commodore 64

B&C

COMMODOR 64 MIKROGÉP

PROGRAMOZÁSA

Belső használatra

Budapest
1983.

1. Gép felépitése, főbb jellemzői

A COMMODORE 64 mikroszámitógép egy univerzális elektronikus,digitális gép. Ez azt jelenti, hogy sok célu elektronikus és digitális jelekkel dolgozó számitógép.

Mikroprocesszor

A gép 8 bites mikroprocesszorának tipusa MOS6510.

1.1. Memória

A gép 64 Kbyte RAM - irható/olvasható memóriával rendelkezik. 1K RAM = 1024 betü vagy jel tárolására alkalmas. Az egész számok /maximális érték ± 32767/ két byte területet foglalnak el. Igy 1K területen 512 db egész számot tárolhatunk. Valós számok esetén egy valós szám ábrázolásához 5 byte szükséges, igy 1K byte-ba összesen 204 valós számot tárolhatunk.
A memóriából a gép saját használatára is felhasznál bizonyos területeket. Ezért nem a teljes 64K memória áll rendelkezésünkre.
Felhasználható terület:

BASIC nyelv használata esetén 38K

gépi nyelv esetén 54K RAM

Igy ha csak adatterületként használnánk a gépet,
a tárolható maximális értékek /BASIC esetén/:

7782	valós szám
19456	egész szám
38912	karakter /betü, számjegy, egyéb jel/.

Emellett a gépben 20 Kbyte ROM, a felhasználó számára nem hozzáférhető terület áll rendelkezésre, amely a BASIC értelmezőt /interpretet/, karakterkészleteket tartalmazza és egyéb feladatokat lát el.

1.2. Képernyő

A géphez tetszés szerinti szines vagy fekete-fehér képernyőt hozzátehetünk, amellyel a TV 21/60 közötti csatornái foghatók. A készülék közvetlenül csatlakoztatható a TV antennájához. A megfelelő beállitás a 36. csatornára való hangolással érhető el.
A képernyőn a következő méretek használhatók:

40	oszlop /betühely/
25	sor /betühely/

Szines TV esetén összesen 256 képernyő/keret szinkombiráció adható meg. A kiirási terület 16 különböző szinüre "festhető".

Ezek a szinek a következők:

alapszinek:

fekete, fehér, piros

türkisz, lila, zöld

kék, sárga

kiegészitő szinek:

világos piros /cinóber/, narancs

barna, 1.szürke, 2.szürke, 3.szürke

világoszöld, világoskék

Ugyanennyi szinre lehet "festeni" a kiirási terület keretét és minden egyes kiirt jelet is.

Kiirható, megjelenithető standard jelek száma:

255 jel

124 ebből a grafikus jel.

Ezenkivül 8 mozgatható tárgy adható meg, amely tetszés szerint szinezhető.

Grafikusan megjelenithető jelek száma:

320 x 200 pont.

Ez az egy megjelenithető betü /számjegy/ hely 8x8-as területre bontásával adható meg. A saját generált jelek mellett a gép összesen 62 grafikus billentyüt is tartalmaz.

1.3. Hangeffektusok

A gép magasfoku zenei képességekkel rendelkezik. A zenei oktávok száma összesen 9. Minden hangot billentyüzetről és programból is megszólaltathatunk.

Összesen 4 féle hullámforma adható meg:

háromszög jel

négyszög jel

szinuszos hullám

zaj /változó/

Ezen kivül változtatható a hangerősség, a hang hosszusága és egyéb jellemzői. A hangeffektusok változtatása a paddle berendezéssel is elérhető.

Különböző zenei eszközök is beállithatók a segitségével.

A szimulálható zeneszámok a következők:

zongora, furulya

csemballó, xilofon

orgona, trombita

A gép HI-FI kimenettel rendelkezik, igy a létrehozott hangeffektusok, zenei számok magnetofonra tehetők.

Három hangzatot lehet a beépitett /un. SID = Sound Interface Device/ IC segitségével lejátszatni együtt vagy külön-külön is. Mivel ezek különböző zenei eszközök is lehetnek, igy a gépet zeneszerzésre is lehet használni. A zene megszólaltatható a TV hangszóróján keresztül, vagy közvetlenül kivihető magnetofonra, erősitőre.

Programozás

A gépen feladatokat megoldani az emberi nyelvhez közelálló BASIC nyelven lehet.
A BASIC nyelven belül a gépi parancsok nagy részét végrehajthatjuk, és bármelyik billentyü "beadható" a programba.

Lehetőség van ASSEMBLER szintü programok irására is, de ekkor egy további software /forditó/ szükséges, amely az ilyen nyelvü programokat leforditja a gép nyelvére. Ez a nyelv meglehetősen nehézkes: az alapmüveletek közül például csak a 2 byte-os összeadást és a kivonást tartalmazza. A fentieken kivül külön megrendelhetők a géphez a következő nyelvek:

BASIC forditó

PETSPEED

UCSD PASCAL

COMAL, LOGO

FORTH

BASIC kiterjesztés

A gépben levő BASIC nyelvet több, mint 100 utasitással bőviti ki a SIMON'S BASIC. Az utasitások közül a fontosabbak a következő csoportokba oszthatók:

- programozást segitő eljárások;
- grafikus utasitások;

- karakter és szellem /sprite/ generáló utasitások
- zenei utasitások;
- hagyományos BASIC utasitások továbbfejlesztése.

CP/M operációs rendszer

A gépet egy további fizikai egységgel kibővitve a CP/M operációs rendszer is rendelkezésünkre áll. Ez azt jelenti, hogy a CP/M operációs rendszer alatt futó programok lefuttathatók gépünkön. Ez az egység egy 280 mikroprocesszort tartalmaz, igy az ilyen gépi nyelvü programok is végrehajtathatók a gépen.

PET emulátor

Egy másik fizikai egységgel bővitve a gépet, a PET software is futtatható a számitógépen. A PET gépre egy sor tervezési, pénzügyi, gazdasági adminisztrációs rendszert is kidolgoztak. Igy a gép felhasználási lehetősége nagyságrendekkel kibővül.

1.4. Kapcsolható perifériák

1.4.1. Kazettás egység

A géphez közvetlenül csatlakoztatható egy, a cég által gyártott kazettás magnetofon. A berendezés BASIC nyelvből és klaviaturáról üzemeltethető. BASIC nyelvből adatokat irathatunk ki, illetve olvastathatunk be a magnetofon segitségével.

Billentyüzetről kazettára a gépbe levő programot irhatjuk ki. Ugyanigy lehetséges a kazettán elhelyezett /tárolt/ programot a gépbe beolvasni. A kazettás egység a szabvány 4.75 cm/sec sebességgel müködik, igy elsősorban programok tárolására használható. Adatok ki/bevitele a fentiek miatt meglehetősen lassu.

Ezenkivül ügyelnünk kell a mechanizmus és az iró/olvasófej tisztaságára.

Szalagként használhatunk speciális adatkazettákat /például VERBATIM R 300 NH adatkazettát/, de megfelelőek a kereskedelemben kapható jobb minőségü zenei célokra használt kazetták is. Célszerü a C 60 kazettákat használni - a szalag megnyulás elkerülésére.

1.4.2 Nyomtató berendezés

A géphez közvetlenül csatlakoztatható a COMMODORE cég mátrixnyomtatója. A géphez a gyár többféle tipusu nyomtatót ajánl. A legegyszerübb és legolcsóbb a VC-1525 tipusu félgrafikus nyomtató. Ezzel másodpercenként 30 jelet tudunk nyomtatni a hazai forgalomban is beszerezhető A3 méretü két oldalán per-

forált papirra. Az egyes A/4 méretü lapok természetesen össze vannak kapcsolva, hasonlóan a nagy gépeken használatos papirokhoz.

A printeren kétféle betüméretet lehet használni. A kisebb betüméretteI 80 jelet lehet egy sorba irni.. A géppel egy eredeti és két másolati példány készithető.

Egyéb javasolt nyomtatók

SEIKISHA GP 100 A

Hasonló teljesitményü a VC-15-25

4,5 kg

ITOH 8510/1550

136/230 oszlop

oda/visszairás

120 cH/sec

8,5/11 kg

COMMODORE 8023

136/250 oszlop

150 cH/sec

11,6 kg

A BASIC nyelvből mint egy önálló perifériát használjuk a nyomtatót. Ugyanigy meg kell nyitni, mint egyéb

egységeket és a szabványos PRINT... utasitások-
kal irathatunk ki a gépre adatokat.

Lehetőség van ezenkivül parancsok kiadására is. Igy kilistáztathatunk a gépben lévő programot is a nyomtatón.

1.4.3 Mágneslemez egység

A géphez sokféle floppy egység és kemény lemezes egység csatolható.

A cég a VIC 1541-es floppy disk egységet szállit-
ja sztenderd egységként. A lemez 5 1/4 zoll át-
mérőjü.

Fizikai felépitése

A lemezegység blokkhossza 255 CH. Kiirás esetén az adatok között elválasztójel is szükséges, amit be kell generálni.

A használható fizikai blokkok száma 655. Igy a lemezkapacitás durván 170 Kbyte.

BASIC-ből használhatjuk a lemezt szekvenciális és random állományok felépitésére. Néhány géppel szállitott program a lemez kezelését, formázását, a rajta lévő anyagok listázását segiti elő.

1.4.4 Egyéb perifériális egységek

Görberajzoló

A mátrixnyomtató csak kevés grafikus lehetőséget tartalmaz. Ezért célszerü rajzolási célra felszerelni a gépet görberajzoló /plotter/ berendezéssel is.

A cég jelenleg kétféle görberajzolót ajánl. Az egyik sikplotter, a másik tekercsplotter.

A sikplotter A3 méretü, a rajzolásra felhasznált terület valamivel kisebb A3-nál.

A tekercsplotter /VX4731/ négyszinü tollal rendelkező egység. Rajzolási terület 287*1000 mm. Maximális sebessége 200 mm/sec /tengelyek mentén/, lépéspontosság 0,1 mm.

IEEE488 csatoló

A csatoló berendezés segitségével a szabványos CBM perifériák a géphez köthetők. Ilyen szabványos periféríák: disk egység:pl.1 és 2 Mbyte-os dual, nyomtató, stb.

Géprendszer felépités

Egy önálló cartridge segitségével a gép egy másik COMMODORE 64-es berendezéssel kapcsolható össze. Ez lehetővé teszi, hogy a perifériákat a gépek közösen használják, és hogy a gépek egymásnak háttérgépként szolgáljanak.

További lehetőség, hogy a gépet egy olcsó COMMODORE modemen keresztül rákössük egy nagyobb számitógépre. Ekkor a gép intelligens terminálként dolgozik.

Fényceruza, egyéb egységek

A géphez közvetlenül köthető fényceruza. Ezzel a képernyőn keresztül végezhetünk bizonyos operációkat /rajzolás, törlés stb./.

A paddle és a joystik a gép grafikus képességeit segitik elő. Bizonyos zenei effektusok is segithetők ezen berendezések segitségével.

2. Gép billentyüzete, csatlakozói

A gép egy irógép billentyüzethez hasonló gombokkal van ellátva. A gép hát- és jobboldalán helyezkednek el a különböző csatlakozási lehetőségek.

2.1 Bekapcsolás

A tápegység a géptől független egység. Ez közvetlenül a hálózatra kapcsolható. Teljesitményfelvétele 30 W. A tápegységből kijövő kábelen lévő speciális 4 kimenetü csatlakozót a gép jobboldalán lévő utolsó helyre kell bedugni /Power felirat/. Emellett helyezkedik el a gép bekapcsolására szolgáló billenő kapcsoló. Ennek ON állása esetén van a gép bekapcsolva, és ekkor a berendezés jobb felső sarkán lévő piros lámpa /POWER feliratu/ kigyullad.

A gép hátoldalának közepén van egy a TV antenna csatlakozójához hasonló csatlakozó. Ezt kell a géphez mellékelt kábellel összekötni a TV antenna-csatlakozójával. A TV készüléket be kell kapcsolni, és a beállitást a 21-60 csatornánál kell a 36-os csatornára behangolni. A bekapcsolás és beállitás után 1-2 másodperccel a gép a következő szöveget irja ki:

```
    **** COMMODORE 64 BASIC V2 ****

64K RAM SYSTEM  38911 BASIC BYTES FREE

READY
```

2.2. Gép csatlakozói

Jobboldali csatlakozók

A gép jobboldalán három csatlakozó és egy kapcsoló helyezkedik el.

Ezek elölről hátrafelé a következők:

1. CONTROL PORT 1
2. CONTROL PORT 2
3. ON/OFF kapcsoló
4. POWER csatlakozó

Az 1. és 2. CONTROL PORT /csatlakozók/ a következő egységek csatlakoztatását szolgálja:

JOYSTICK /botkormány/

Fényceruza

PADDLES

A csatlakozók kiosztása a következő:

CONTROL PORT 1:

Csatlakozó száma	Jel	Megjegyzés
1	JOYA0	
2	JOYAA1	
3	JOYAA2	
4	JOYAA3	
5	POTAY	
6	Gomb A/LP	
7	+5V	max 100 mA
8	GND	
9	POTAX	

CONTROL PORT 2:

1	JOYB0	
2	JOYB1	
3	JOYB2	
4	JOYB3	
5	POT BY	
6	Gomb A/LP	
7	+5V	max 100 mA
8	GND	
9	POT BX	

Megjegyzések

a./ A fenti táblázatokban a csatlakozók száma:

felső sor jobbról balra 1-5

alsó sor jobbról balra 6-9

b./ A POTAY, POTBY a Paddle potenciométerét

röviditi

c./ A Gomb A/LP a fényceruza illetve a Joystick gomb-

jait jelölik

Az ON/OFF kapcsoló a gép ki/be kapcsolására szolgál.

A legutolsó 4. számu csatlakozó /POWER/ a tápegység csatlakozója.

Hátoldali csatlakozók

A gép hátoldalán balról jobbra összesen 6 csatlakozó van,

Ezek a következők:

1, Modul csatlakozó

2, TV csatlakozó

3, Audió/video kimenet

4, Soros kimenet

5, Magnetofon csatlakozó

6, Felhasználói /user/ csatlakozó.

1. A 22 pólusu modul csatlakozó a különböző program és játékmodulok géphez való kötésére szolgál. A kiosztás megadásától eltekintünk, azt a gépkönyv tartalmazza.

2. TV csatlakozó

   A géphez adott kábellel ezt a csatlakozót kell összekötni a TV antennacsatlakozóval /75 Ohm/

3. Audió/video kimenet

   A gép innen közvetlenül magnetofonhoz vagy erősitőhöz csatlakoztatható /HIFI kimenet/ Ugyanez a kimenet szolgál vedojel képernyőre vitelére is.

   A csatlakozó kiosztása a következő:

   1. fényerősség
   2. GND /föld/
   3. AUDIO kimenet
   4. VIDEO kimenet
   5. Audió bemenet

Megjegyzés:

a./ Számozás a gépkönyvben megadott számozással megegyezik

b./ Az 1-3 csatlakozóval csatlakozhat például magnetofonra,igy a gép által szolgáltatott zenét felvehetjük közvetlenül.

4. Soros kimenet

   Ezen a csatlakozón keresztül köthetünk a géphez mátrixnyomtatót vagy disk egységet.

A csatlakozó kiosztása a következő:

1. Soros SQR bemenet
2. GND /föld/
3. Soros vezérlő jel ki/be /ATN IN/OUT/
4. Soros óra ki/be /CLK IN/OUT/
5. Soros adat ki/be bitenként
6. Nincs csatlakoztatva

Megjegyzések

a./ A csatlakozók számozása megegyezik a gépkönyvben megadott számozással

b./ Több egység egymáshoz kötve csatlakoztatható a berendezéshez.

5. Magnetofon csatlakozó

Ebbe a csatlakozóba kell bedugni a géphez adott magnetofon csatlakozóját.

A csatlakozó kiosztása a következő /számozás jobbról balra/

1. GND /föld/
2. +5V
3. Kazettás egység motorja
4. Kazetta olvasás
5. Kazetta irás
6. Kazettás egység kapcsolója

6. Felhasználói /user/ csatlakozó

Szabadon programozható ki/bemenet és steckmodulok csatlakozási lehetősége. A csatlakozó kiosztását kézikönyben találjuk meg.

2.3. Billentyüzet

Az összes billentyü kombináció jelentésétől eltekintünk,mivel azt a későbbi tárgyalásnál az egyes témaköröknél elmondjuk. A fontosabb billentyük megadását az alábbiakban tesszük meg.

SHIFT billentyü

Az irógéphez hasonlóan alsó/felső állás billentyü. Bekapcsoláskor a gép a nagybetüket és az alsó álláshoz tartozó jeleket értelmezi. A SHIFT lenyomásával a billentyük elsö lapján feltüntetett jobb oldali grafikus jeleket adja /például A lenyomása esetén fekete sziv!/. A SHIFT és a "commodore" billentyü /bal alsó sarok/ együttes lenyomásával a billentyüzet kis betüre vált, de az egyéb jelek továbbra is felsö állásban maradnak /pl. számjegyek/. Ezen billentyük együttes ujra lenyomásával térhetünk vissza az eredeti helyzetbe.

Commodore

A Commodore billentyü a bal oldali grafikus jelek választására szolgál. Ezenkivül az alapszinskála bővitését is ezzel a billentyüvel érjük el.

CTRL billentyü

A billentyü az alapszinek megadására szolgál. A billentyük lenyomása által kinyomtatott jelek szinét a CTRL és valamelyik szin billentyü lenyomásával változtathatjuk.

Az alapszinek billentyüi /CTRL + billentyü/:

1: fekete	2: fehér	3: piros	4: türkiz
5: lila	6: zöld	7: kék	8: sárga

Az RVS ON billentyüt lenyomva a CTRL-lel együtt a karakter háttérszine és a jel szine felcserélődik. Ezt cseréli vissza az RVS OFF billentyü /9 és 0 billentyük/.

A CTRL billentyü másik funkciója a program listázásának lassitása.

RVS ON/OFF

Az ON billentyü a fenti funkció mellett a grafikus jelek inverzének /forditott szinképének/ kirajzolására szolgál. A CTRL-lel együtt megnyomva látja el ezt a funkciót. Az OFF billentyü váltja vissza az eredeti funkciót.

RUN/STOP

A billentyüvel a gépben lévő program futását megállithatjuk, ha a program fut. Ha a SHIFT billentyüvel együtt nyomjuk meg, akkor a következő programot a gép betölti kazettáról és a program végrehajtását is elkezdi.

CRST billentyük

A jobboldali billentyü a kurzor /villogó négyszög/ jobbra /SHIFT billentyüvel balra/ való mozgatását végzi.

A baloldali billentyü a kurzort lefelé /SHIFT billentyüvel felfelé/ viszi. Ennek szerepe a már begépelt adatok, program módositásánál, javitásánál van.

RETURN

A billentyü a képernyőre begépelt jeleket viszi be a gépbe.

RESTORE

A billentyüt a RUN/STOP billentyüvel együtt megnyomva a gépet alaphelyzetbe viszi, mintha ujra ki/be kapcsoltuk volna. A gépben levő program viszont megőrződik és tovább tudunk vele dolgozni. Különösen hasznos ez, ha valamilyen okból "szétesik" a képernyő, azaz beazonosithatatlan jeleket ir ki, és nem tudjuk tovább felügyelni.

INST/DEL

Javitásra szolgáló billentyü. Alapesetben törli a kurzornál lévő karaktert. SHIFT billentyüvel a kurzortól jobbra levő jeleket egy hellyel jobbra tolja, igy egy jelet beszurhatunk.

CLR/HOME

Lenyomva a kurzor a képernyő bal felső poziciójába megy, de a képernyő tartalma változatlan marad. A SHIFT billentyüvel együtt a képernyőt is kitörli.

f1 - f8 billentyük

A gép jobb oldalán levő 4 billentyü 8 gépi kódu programot /programrészletet/ rendelhet a billentyükhöz. Használható ezenkivül BASIC programban is, mivel mindegyiknek saját "jelentése" van.

Grafikus jelek

Beirhatók a programba, igy rajzolási funkciókat látnak el.

Megjegyzés

A billentyük nagy része "beirható" a programba. Végrehajtásuk ugyanazt a szerepet játsza, mintha magát a billentyüt nyomtuk volna le. Igy például a CRL/HOME billentyüt is "beirhatjuk" egy PRINT utasitásba, aminek hatására futás közben a képernyő aktuális tartalma kitörlődik, Hasonlóképpen szinezhetjük futás közben a kiirandó adatokat, szövegeket is.

3. BASIC programok irása

A továbbiakban a BASIC nyelv COM.64-en értelmezett változatával ismerkedünk meg.

Program formátuma

A BASIC programot kétféle módon irhatjuk: közvetlen végrehajtásu és tárolt program formájában.

Közvetlen végrehajtásu program esetén az utasitást beütjük a billentyüzetről, és a RETURN gomb lenyomása után az utasitás azonnal végrehajtódik. Ekkor a gépet a zsebkalkulátorokhoz hasonlóan használjuk: az utasitást a gép nem tárolja el a memóriájában.

A tárolt programok irása a billentyüzetről történik. A programot utasitásonként gépeljük be a gépbe, és a RETURN billentyüvel lehet tároltatni a memóriába. Az ilyen program csak akkor hajtódik végre, ha a végrehajtásra külön utasitást adunk /a RUN szó begépelésével/. A programozás lehetőségeivel ismerkedünk meg a továbbiakban.

Tárolt programok irása

Ha az utasitássorozatot el akarjuk tenni a memóriába, akkor minden egyes utasitást vagy utasitás-

csoportot egy számmal /un. cimkével/ kell ellátnunk. A továbbiakban áz utasitás vagy utasitássorozat ezzel a számmal azonositható.

Ha az utasitás tul hosszu, és nem fér ki egy sorba, akkor azt a következő sorban folytatjuk. Egy utasitás felirásához legfeljebb két sort használhatunk fel.

Az utasitás formája ekkor a következő:

$$c \quad u_1 : u_2 : u_3 : \ldots : u_n$$

ahol c az utasitást vagy utasitáscsoportot azonositó cimke /0 ≤ c ≤ 65999/

$u_1, u_2, \ldots, u_n$ a c cimkével megjelölt utasitássorozat,

Szabályok a program irásával kapcsolatban:

- ha a c cimkéhez több utasitás tartozik, akkor az utasitásokat egymástól : /kettőspont/ választja el;
- egy cimkével jelzett utasitás /utasitáscsoport/ összesen legfeljebb két sorból állhat;
- az utasitás a RETURN billentyü lenyomása után tárolódik el,
- ha már volt c cimkéjü utasitás korábban eltárolva, akkor az előző utasitás törlődik;
- a gép az utasitásokat automatikusan növekvő sorrendben rendezi;

- a cimke és az utána következő utasitás között akárhány üres helyet irhatunk, a gép ezeket figyelmen kivül hagyja és ugy tárolja, mintha a cimke után egy üres leütés lenne.

A fenti szabályokból is látszik, hogy a RETURN billentyü lenyomása előtt az utasitást lehet módositani, javitani.

Megjegyezzük, hogy a sorszámozást célszerü tizesével végezni, mivel ekkor a már elkészült programba ujabb utasitásokat be tudunk helyezni a megfelelő sorszám megválasztásával. Jegyezzük meg, hogy a programirás során sehol sem szükséges szóközöket tenni.

Példa:

```
10IFQ=1THENGOTO5
```

A programba tetszésszerinti helyre kommenteket /megjegyzéseket/ is beépithetünk, amely a további felhasználás, módositás szempontjából lehet fontos.

Ezt a következő formában végezzük:

```
c  REM  t
```

ahol c utasitáscimke

REM a komment utasitás alapszava

t tetszésszerinti, a gép által megengedett karaktersorozat.

A továbbiakban a BASIC programnyelv utasitásait és egyéb, a programmal kapcsolatos utasitásokat ismertetünk.

4. BASIC alapelemei

4.1. BASIC alapjelek

A BASIC programban csak bizonyos jelek fordulhatnak elő, amiket alapjeleknek nevezünk.

Az alapjelek a következők:

- az angol ABC kis és nagybetűi:

  A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T,V,W,X,Y,Z

- írásjelek:

  !,",?,,,.,-;,:,( )

- reláció jelek:

  < > =

- aritmetikai müveleti jelek:

  +,-,*,/,

- egyéb jelek:

  $, //, %, &,

4.2. Konstansok

A programban az utasitások számokat is tartalmazhatnak /pl. aritmetikai, logikai kifejezések stb/.

Ezeket a számokat hivjuk konstansoknak. A matematikához hasonlóan itt is megkülönböztetünk egész és valós konstansokat. Ezenkivül használhatunk karakter /un. Hollerith/ konstansokat is.

4.2.1. Egész és valós konstansok

Egész konstansok

Egész tipusu konstans + vagy - előjelből és számjegyekből áll. Amennyiben a szám pozitiv, ugy a + jelet nem szükséges kitenni.

Az egész tipusu konstans legnagyobb és legkisebb értékei:

$$-32768 \leq x \leq 32767$$

Példák:

Helyes	Helytelen	
67	"67"	/nem egész tipusu!/
-13582	621651	/tul hosszu/
10	61.5	/tizedes pont!/

Valós konstans

Valós konstans előjelből, egész és törtrészből és kitevőből állhat.

Az alábbi valós konstans formátumok megengedettek:

± n.m	± n.mE±s
	± n.E±s
± .m	± .mE±s
± nE±s	

ahol n az egész részt, m a törtrészt, s pedig a kitevő reprezentáló szám, és m,n,s egész konstans /matematikai értelemben, tehát a % jel nem kell a számok után!/.
A fenti listából látszik, hogy két alak kivételével mindegyik tartalmaz tizedespontot. A ±nE±s alakban viszont kitevő van.

A + előjel mind a szám,mind a kitevő elől elmaradhat, ha ezek pozitivak.

A kitevő értelmezése a következő: az előtte megadott számot 10 annyiadik hatványával kell megszorozni, amennyi az E után áll. Igy az $n_1.m_1.m_2.m_3$ E + s_1s_2 konstans az alábbi számmal egyenlő

$$n_1m_1m_2m_3 * 10^{s_1s_2}$$

Az alábbi valós számok mindegyike ugyanazt a numerikus értéket képviseli

3.1415

0.3115 E1

31415 E-4

31.415 E-1

A valós számok értéke az alábbi határok között lehet:

± 1.70141183E+38, ± 2,93873588E-39

A konstansok a számitógép memóriájában kerülnek elhelyezésre. A fenti minimális, maximális értékhatárok igy a tárolóhely kapacitásával kapcsolatosak.

A számitások eredményei 9 számjegy pontossággá kerülnek kiirásra.

Az értékek kiirása kétféleképpen történik:

- ha a szám 9 jegynél rövidebb, azaz 10^{10} és 10^{-10} közé esik, akkor kitevő nélküli formában;
- ellenkező esetben kilenc értékes jegy és kitevős alak ugy, hogy az értékes jegyeket a /-10, 10/ intervallumba transzformálja, és kitevővel irja ki.

Példák:

Érték a számitógépben	Kiirás
15	15
12345,123456	12345.1235
12345.E24	1.2345E+28
99999,9999E25	9.99999999E+29
0.09876543E36	9.876543E+34
0,000000001E-16	1E-25
-0,2345678E-26	-2.345678E-27

4.2.2. Karakter konstans

Karakter /un.Hollerith/ konstanst a programban kiirásra, összehasonlitásra stb. használjuk. Karakteres konstansként a gép tetszésszerinti jelei használhatók. Elejét és végét idézőjel jelöli. Maximális hossza megegyezik egy utasitás maximális hosszával.

Példák:

"ALMA"

"MOST 98 C A HOMERSEKLET"

4.3. Változók

Változók a matematikában megszokott fogalmat takarnak, a programban is értéket vehetnek fel. A program futtatása során minden változónak megfelel egy memóriaterület, ahová a változónak adott /számitott vagy beolvasott/ érték kerül. A változókat a matematikához hasonlóan nevekkel azonositjuk

- legfeljebb 2 jel hosszuságu,
- az első jel az angol ABC nagybetűinek valamelyike
- a második jel egy számjegy /0-9 valamelyike/, vagy
  egy betü

Példák:

A1, B0, C3, Q9, AB, CQ

Három adatnév van lefoglalva a VIC BASIC-ben. Ezek: ST, TI, TI$. Az ST az olyan I/O müveletekkel kapcsolatos változó, amely a lemezzel, szalaggal kapcsolatos.

A TI$ változóban lehet az órát, percet és másodpercet beadni a következő formában "hhppmm", ahol hh az órát, pp a percet, mm a másodpercet jelöli. A TI változó a belső óra idejétől függő változó, és minden 1/60-ad másodpercben nő az értéke. Kiindulási értéke a bekapcsoláskor nulla. Ezt az értéket változtathatjuk meg a TI$ változó értékének beállitásával.

4.3.1. Változók tipusa

A konstansokhoz hasonlóan a változók is három tipusuak lehetnek:

- egész,
- valós,
- karakter

Egész változót a változó neve után irt % jellel adunk meg.

Karakter változót a változónév után irt $ jellel látnak el. Maximális hossza 256 jel.

Valós változó minden olyan változó, amely nem a fenti feltételek valamelyikének felel meg. Azaz minden változó valós, amely után % vagy $ jeleket nem irtunk.

Változók deklarációja

Változók deklarációja alatt a változó tipusának megadását értjük. Mint fent láttuk, egész, valós és karakter tipusu változó tipusát külön nem kell megadni, mivel azt a gép a változó első előfordulásakor automatikusan deklarálja.

4.3.2. Tömbök megadása

Feladatok megoldása során gyakran van szükségünk vektorokra, mátrixokra. Az ilyen tipusu változókat nevezzük tömböknek vagy indexes változóknak. Indexes változók maximálisan 255 dimenziósak is lehetnek.

Egy indexes változó egymás utáni helyeket foglal el a memóriában, az alábbi szabályok szerint:

- vektor elemei egymás utáni memóriahelyre kerülnek;
- mátrix elemei oszlopfolytonosan helyezkednek el.

Tömböket deklarálni szintén a DIM utasitással lehet.

Az utasitás formája a következő:

$$\text{DIM } vn_1(d_1),\ vn_2(d_2),\ldots,vn_m(d_m)$$

ahol vn_i az i-edik tömb neve;

d_i az i-edik változó méretének /méreteinek/ maximális száma plusz 1, ugyanis az indexek nullától indulnak.

. A vn_i változó lehet egész, valós vagy karakter tipusu. Ekkor egész tipusu változó esetén % jelet, karakter tipusu esetén pedig $ jelet kell tennünk a változó név után. Mivel valós változó tipus a változó alapértelmezése, igy az ezen nevek után nem kell semmit sem irnunk.

A d_i annyi egész számból állhat, ahány dimenziós a tömb. Ha a tömb egydimenziós /vektor/, akkor a változó név után irt egész szám a vektor elemeinek számát adja meg. Ha mátrixot adunk meg, akkor a d_i két pozitiv számból áll, a számok vesszővel vannak egymástól elválasztva. Az első szám ekkor a mátrix sorainak, a második az oszlopainak a számát jelenti. A d_i ezenkivül egész tipusu változó is lehet.

Példák:

a./ DIM A(10), B% (25), C1(30, 10)

b./ DIM X(10,2), X1% (40)

A fenti példában az a. esetben A egy valós 11 elemü vektor, B egy 25 elemü egész vektor, C1 egy 31 sorból és 11 oszlopból álló valós mátrix; a b. esetben X egy 11 sorból és 3 oszlopból álló valós mátrix, X1 egy 41 elemből álló egész vektor.

Megjegyezzük, hogy a gépnél a vektorok, mátrixok megadása nem mindig kötelező: ha a DIM utasitást nem használjuk, és valamelyik változó után indexet /indexeket/ irunk, akkor a gép ugy veszi, mintha 11 elemü vektor /mátrix/ lett volna deklarálva a program elején.

Példák:

1. A = B(2) + B(8)

2. C(6,8) = B(3) * X

A fenti példában a B vektorral és C mátrixszal tudunk deklarálás nélkül is dolgozni, mivel a gép az 1. utasitás végrehajtása közben egy 11 elemü vektornak, a 2. utasitás végrehajtása után egy 11 sorból és ugyanannyi oszlopból álló mátrixnak foglal helyet. Igy a fenti utasitások végrehajtása során egy

DIM B(10),C(10, 10)

deklarációnak megfelelő müveleteket végez.

4.4. Aritmetikai kifejezés és értékadó utasitás

4.4.1. Aritmetikai kifejezés felépitési szabályai

Az aritmetikai kifejezés a BASIC nyelvben megegyezik a matematikában szokásos fogalommal:

Aritmetikai müveleti jelekből /operátorok/ és aritmetikai elemekből /operandusok/ áll. Az operandusok és a kifejezés értéke egész vagy valós szám.

Felépités szabályai is megegyeznek a matematikában ismert szabályokkal, különbség csak a felirási módban, müveletek jelölésében van.

Aritmetikai müveleti jelek a következők:

Müveleti jel	Müvelet
+	összeadás
-	kivonás
*	szorzás
/	osztás
↑	hatványozás

Kifejezésekben a következő aritmetikai elemeket használjuk:

konstans

egész vagy valós változó /egyszerü vagy indexes/

függvény kifejezés

logikai kifejezés

Példák:

Matematikai	BASIC
$a + b$	A + B
$a - 3b^2$	A - 3 * B ↑ 2
$a + b^2$	A + B ↑ 2
$\frac{a}{a + b}$	A/ (A + B)

Az aritmetikai kifejezés fogalmát általánosabban használjuk, mint a matematikában, ugyanis egy aritmetikai elem önmagában is aritmetikai kifejezés.

Az aritmetikai kifejezéssel kapcsolatos szabályok:

1. Felépités általános szabálya:

   Aritmetikai kifejezés lehet aritmetikai elem vagy két aritmetikai elem müveleti jellel összekapcsolva.

2. Ha előjeles szám előtt müveleti jel is van, azt nem fontos zárójelbe tenni.

   Példa:

   X+Y /-3*-5

   A * -B

   A * A/ -C

3. Ha X, Y aritmetikai kifejezések, akkor az alábbiak szintén kifejezések:

X	X + Y	X - Y	X/Y
X * Y	X	Y	- X

4. A müveletek végrehajtásának sorrendje a matematikában szokásos sorrend, azaz:

   - elsőként a logikai kifejezéseket és függvények értékeit számoljuk ki,
   - majd a hatványozást végezzük el.
   - ezután a szorzásokat és osztásokat,
   - legvégül az összeadásokat és kivonásokat.

   Az azonos sorrendü müveleteket balról jobbra haladva egymás után végezzük el.

5. Csoportositás, vagy müveleti sorrend megváltoztatása céljából zárójelet használunk, de ügyelni kell a hatványozásra és szorzásra:

$(x + y)^3$	helyesen	(x + y) ↑ 3
(x + y)(z + w)	-"-	(x + y) * (z + w)

A zárójelet tartalmazó aritmetikai kifejezésben először a zárójelben levő müveleteket végezzük el.

6. A hatványozás maga is kifejezés, azaz X ↑ (1+2) is az. Vigyázzunk a müveletek sorrendjére hatványozás esetén. A hatvány definiciója miatt A ↑ B ↑ C a következő sorrendben kerül végrehajtásra:

a./ Z = B ↑ C /Z részeredmény/

b./ W = A ↑ Z /W a kifejezés értéke/

4.4.2. Aritmetikai kifejezés tipusa

Az aritmetikai kifejezések tipusát a következőképpen határozzuk meg:

- egész tipusu az aritmetikai kifejezés, ha benne minden aritmetikai elem /konstans, változó és függvénykifejezés/ egész,
- valós tipusu, ha legalább egy aritmetikai elem valós.

Összefoglalva: a kifejezés tipusát a domináns /legmagasabb/ tipusu változó határozza meg.

A változó tipusok hierarchiája a magasabbtól az alacsonyabb felé:

valós,

egész.

A kevert tipusu aritmetikai kifejezés esetén /azaz amelyikben nemcsak azonos tipusu elemek vannak/ az aritmetikai elemeket a gép azonos tipusuvá alakitja át a müvelet elvégzése előtt. Ezt nevezzük konvertálásnak. Az átalakitás során az alacsonyabb hierarchiáju kifejezéseket alakitja át a gép a kifejezésben szereplő magasabb hierarchiáju tipusuvá! Igy ha egész és valós elemek szerepelnek egy kifejezésben, akkor a müvelet elvégzése előtt az egész elemek valós tipusuvá konvertálódnak.

Megjegyezzük, hogy karakter tipusu kifejezésekre értelmezve van az "+", amely ez esetben a két konstans vagy változó értékének egymás mellé irását jelenti.

4.4.3. Értékadó utasitás

Az aritmetikai értékadó utasitás egy olyan egyenlőség, amelynek egyik oldalán egy változó neve, a másik oldalán egy aritmetikai kifejezés szerepel.

Az értékadó utasitás két formája ismeretes a számitógépen. Az egyik a LET "szabvány" nyelvi utasitás, a másik más programozási nyelvekben is használatos értékadó utasitás.

A LET utasitás a következő alaku

LET v = e

ahol v egy egyszerü vagy indexes változó

e aritmetikai kifejezés.

Az utasitás végrehajtása során kiszámoljuk az e aritmetikai kifejezés értékét és ezt rendeljük a v változóhoz.

Az értékadás során ha a v = e kifejezés két oldala tipusban nem egyezik meg, ugy az e értéket a v tipusára konvertáljuk. A konvertálási szabályt a fejezet végén mondjuk el.

Megjegyezzük, hogy

- ha v karakter tipusu, akkor a jobboldalnak is ilyen tipusunak kell lenni;
- a BASIC-ben szokásos LET $v_1 = v_2 = \ldots = v_n$ = e kifejezés hatására a gép semmit sem csinál, és v_i értéke nulla /i=1,2,...,n/.

A másik, egyszerübb alaku értékadó utasitást a következőkben ismertetjük.

Általános formája:

v = e

ahol v egy változó, e pedig egy aritmetikai vagy logikai kifejezés /konstans, változó vagy összetett kifejezés/.

Az utasitás végrahajtása a következőképpen történik:

- elvégezzük a kijelölt müveleteket az e kifejezésben /ha vannak/;

- v értékének a kiszámolt e értékét adjuk.

Mint a fentiekből is látjuk, ez az utasitás nem azonos a matematikai értelemben vett egyenlőséggel. Az A = A - 1 kifejezésből ugyanis a matematikában az következik, hogy -1 = 0! Itt az utasitás végrehajtási szabályai szerint először kiszámoljuk az A - 1 értéket, majd az A változó ezt az értéket kapja. Kiolvasni is másképpen kell az utasitást. Ebben az esetben a következőképpen: A értéke legyen egyenlő A - 1-gyel, vagy általánosan a v értéke legyen az e kifejezés értéke.

Példák:

```
A = - A
B(I,4) = SIN(I) + B ↑ 2
I = J ↑ 2 + A/B
```

Különböző tipusu értékadás esetén konverzió történik. Azaz, ha a v és e tipusa különböző, akkor az aritmetikai kifejezés értéke konvertálódik a bal oldal tipusához.

Az alábbi konvertálási szabályokkal történik a hozzárendelés:

v tipusa	e tipusa	művelet
egész	egész	értékadás
egész	valós	tizedes részt levágva értékadás
valós	egész	egészt valósba konvertálva, értékadás
valós	valós	értékadás

4.5. Reláció és logikai kifejezés

4.5.1. Reláció

Logikai kifejezés előtt a relációkkal kell foglalkoznunk, amelyek a logikai kifejezések legegyszerübb tipusai.

A reláció felépitése a következő:

$$kif_1 \ reop \ kif_2$$

ahol kif_1 és kif_2 vagy aritmetikai kifejezések
/beleértve a konstans és változót is/, vagy
karakter tipusu kifejezések
reop reláció jel

A reláció jelek a következők lehetnek a BASIC-ben:

< kisebb
<= kisebb vagy egyenlő
> nagyobb
>= nagyobb vagy egyenlő
= egyenlő
<> nem egyenlő

A reláció kiértékelése a következőképp történik:

- kiszámoljuk a jobb és baloldali kifejezés értékét;
- összehasonlitjuk a két értéket a reláció jelnek megfelelően;
- ha a feltétel teljesül, akkor a reláció értéke igaz, egyébként hamis.

Ha a kifejezések karakter tipusuak, akkor az összehasonlitás karakterenként történik, mégpedig a karakterek ASCII kódjának numerikus értéke szerint. Ha a két érték nem egyforma hosszuságu, akkor ugy hasonlitjuk össze, mintha a rövidebb szóközzel lenne feltöltve.

A reláció numerikus értéke -1 ha igaz, nulla, ha hamis. Igy aritmetikai kifejezésben is használhatjuk aritmetikai elemként.

A relációk kiértékelésére az alábbi táblázatban mutatunk néhány példát

Reláció	A=1 B=2	A=2 B=2	A=2 B=1	A=0 B=0	
A > B	0	0	-1	-1	0 hamis
A < B	-1	0	0	-1	-1 igaz
A = B	0	-1	0	-1	
A + B <= A * A	0	-1	0	-1	
B * B > A - B	-1	-1	0	-1	
A * A <> B * B + B	-1	-1	-1	0	
A * A + B * B >= /A + B/ ↑ 2	0	0	0	-1	

4.5.2. Logikai kifejezés

Logikai kifejezés értéke a relációhoz hasonlóan lehet igaz vagy hamis. A kifejezés relációkból épül fel, amelyek logikai müveleti jelekkel vannak összekapcsolva.

Logikai kifejezés lehet:

- reláció
- $r_1 L_1 r_2$
- $L_2 r_3$

ahol r_1, r_2, r_3 reláció vagy logikai kifejezés

L_1, L_2 logikai müveleti jel

Logikai müveleti jelek lehetnek:

L_1: AND konjunkció /"és"/

OR diszjunkció /"vagy"/

L_2: NOT negáció /"nem"/

A logikai müveleti jelek "igazság" táblázatai a következők:

A AND B

A \ B	1	0
1	1	0
0	0	0

A OR B

A \ B	1	0
1	1	1
0	1	–

A logikai kifejezés numerikus értékével kapcsolatosak megegyeznek a relációnál elmondottakkal.

Ha A igaz, NOT A hamis; ha A hamis, NOT A igaz.

4.6. Input-output utasitások

Az input utasitásokkal a számitógép memóriájába lehet adatokat beolvasni a billentyüzetről, mágnesszalagról vagy mágneslemezről.

Az output utasitások adatokat irnak ki a képernyőre, mágnesszalagra, lemezre vagy printerre. Az egyszerüség kedvéért a fejezetben csak a legegyszerübb esettel foglalkozunk: a billentyüzetről visszük be az adatokat, illetve a képernyőre iratjuk ki.

4.6.1. Beolvasó utasitások

A továbbiakban több tipusu beolvasó utasitással foglalkozunk. Ezek: adatbevitel billentyüzetről, illetve DATA utasitással megadott konstans listából.

4.6.1.1. INPUT utasitás

Az INPUT utasitás egy vagy több adatot olvas be a képernyőről. A képernyőre ezeket az adatokat a billentyüzettel irjuk be.
Az utasitás formája a következő:

INPUT v_1, v_2,..., v_n

ahol v_i a beolvasandó adatok listája.
A v_i változók egész, valós vagy karakter tipusuak lehetnek.

Az utasitás a következőképp hajtódik végre:

- az utasitás végrehajtásának kezdetén ?-t ir ki a gép;
- ezek után vesszővel elválasztva annyi változót kell beütni a képernyőre, ahány a beolvasandó adatok listáján szerepel;
- megnyomva a RETURN gombot a képernyőn levő első adat értékét kapja a v_1 változó, a 2. adat értékét a v_2 változó ... az n értékét pedig a v_n változó

Megjegyezzük, hogy:

- Az adatokat egyenként is megadhatjuk ugy, hogy minden egyes adat beütése után megnyomjuk a RETURN gombot, Ekkor az első adat beütése előtt egy kérdőjelet, a többi adat beütése előtt két kérdőjelet ir ki a gép. Ezt addig végzi, ameddig a megfelelő számu adatot be nem vittük a gépbe.
- Ha karakter tipusu adat is szerepel a beolvasandó adatok listáján, akkor az adatokat nem kell idézőjelbe tenni.
- Ha egész tipusu változóba viszünk be valós tipusu adatot, akkor a tizedes részt a gép automatikusan levágja.
- Ha több, vesszővel elválasztott adatot gépelünk be a gépbe, akkor az adatlistán szereplő változók számának megfelelő adatot olvas be a gép, a többit figyelmen kivül hagyja és kiirja a következő üzenetet

EXTRA IGNORED

A következő példákban az R betü a RETURN billentyü megnyomását jelöli.

Példák:

```
2     INPUT A$ , B, C%
```

A fenti programrészletet a gép különböző adatok esetén a következőképp hajtja végre:

	Képernyő tartalma		Számitógépbe került eredmény		
a./	? 1,2,3	R	A$	B	C%
			1	2	3
b./	?ALMA,2.1,3.8	R	ALMA	2.1	3
c./	?1	R	1	2	8
	??2	R			
	??8	R			
d./	? A,B,5	R	Hibajelzés /REDO FROM START/		

A tömbök beolvasását ciklusban /az utasitást lásd később/ kell végezni. Ha ugyanis egy tömbnév szerepel a beolvasandó adatok listáján, akkor a tömbbe semmilyen adat bevitel nem kerül. Ugyanis a gép a tömb nevét, ha nem szerepel index utána, egy külön változónak tekinti.

Ha a beolvasandó adatok listáján egy tömb elem szerepel, akkor viszont a beolvasás során csak ez az elem kap csak értéket.

Tekintsük az alábbi példát:

```
1  DIM A(6)
2  INPUT A,A(6)
```

Képernyő tartalma		Adatok értéke a végrehajtás után
		A,A(6), A(1)...A(5)
?2,8	R	2 8 , 0 ... 0
?105,1.5E2	R	105,150, 0 ... 0

4.6.1.2 Egy karakter vagy karaktersorozat gépbe vitele

Egy jel gépbevitelére alkalmas a GET utasitás.Formája

GET v

ahol v a változónév, ahova tárolásra kerül ez a karakter. A leütött billentyü karakterképe nem jelenik meg a képernyőn és kérdőjelet sem ir ki a gép. Ha billentyüt nem nyomunk meg, akkor a gép folytatja a futást, mintha nulla lenne a változó értéke, ha numerikus,illetve space, ha nem numerikus. Ezért sokszor egy ciklussal szokás a kivánt karaktert vagy számot beolvasni.

4.6.1.3. Beolvasás a READ utasitással

A READ utasitással adatokat az előzőleg programba irt DATA utasitást felhasználva tudunk beolvasni. Ezért először ezt az utasitást tárgyaljuk.

A DATA utasitás nem végrehajtható utasitás, hanem un. deklarációs utasitás. A programba akárhol elhelyezhetünk DATA utasitást.

Formája a következő:

DATA $a_1, a_2, \ldots, a_n$

ahol a_i, i=1,2,...,n egész, valós vagy karakter konstansok.

Egy programban több DATA utasitás is szerepelhet. Ezek az utasitások az utasitás előtt álló utasitás cimkével azonosithatók.

A READ utasitással a DATA-val megadott konstans értékeket vihetjük be változókba.

Az utasitás alakja a következő:

READ $v_1, v_2, \ldots, v_n$

ahol v_i /i = 1,2,...,n/ az i-edik beolvasandó egyszerü vagy indexes változó.

Az utasitás a következőképp hajtódik végre:

- a gép megkeresi az utoljára használt DATA utasitás utoljára beolvasott eleme után következő konstanst;
- a v_1 változó ezt az értéket, a v_2 változónak a DATA listán szereplő következő elemet ... adja a gép.

Megjegyezzük hogy:

- ha a DATA lista hosszabb, mint a beolvasandó adatok listája, akkor semmi probléma nem adódik;
- ha a beolvasandó adatok listája hosszabb az összes soron következő DATA listánál, akkor a gép hibajelzéssel megáll.

Példák:

```
1    DATA 1,2,3,4,5,6,7
2    READ A,B
3    DATA 8,9,10,11
4    READ C,D,E,F,H,I,J
5    READ A,B
6    READ C
```

A programrészlet végrehajtása a következőképp történik:

- A 2 sorszámu utasitás hatására az A = 1, B = 2 értékadás játszódik le. A soron következő DATA lista elem a 3., ennek értéke 3.
- A 4 utasitás végrehajtása ekvivalens a C = 3, D = 4, E = 5, F = 6, H = 7, I = 8, J = 9 utasitások végrehajtásával.

- Az 5 utasitás végrehajtása után A értéke 10, B értéke 11.
- A 6 utasitást a gép nem tudja végrehajtani, mivel a DATA listák elemei elfogytak, igy a gép OUT OF DATA hibajelzéssel leáll.

```
1  READ A $, S,D
2  DATA "ABCD", 25, 16.2, 5, "KUTYA",1
3  READ F,G $
4  READ X,Y
5 DATA 2
```

Program végrehajtása:

- Az 1 utasitás megegyezik az A $ = "ABCD", S = 25 D = 16.2 utasitások végrehajtásával
- A 3 utasitás ekvivalens az F = 5, G $ = "KUTYA" utasitásokkal
- A 4 utasitás megegyezik az X = 1, Y = 2 utasitások végrehajtásával.

Mint azt láttuk egy beolvasott DATA elemet többet nem tudunk felhasználni eddigi ismereteink alapján.
A DATA elemek ujra felhasználását segiti a RESTORE utasitás.
Ennek formája a következő:

RESTORE

Ennek hatása a következő:

- a következő READ utasitás a programban levő sorrendben első DATA utasitásra vonatkozik

Példa:

```
1  DATA 25, 42
2  READ A, P, Q
3  DATA 45, 33, 2, 1
4  RESTORE
5  READ S, T
6  RESTORE
7  READ X, Y
```

A fenti programrészlet a következőképp kerül végrehajtásra:

- A 2 utasitás hatására az 1 cimkéjü DATA utasitásból leolvassa az A és P változók értékét, a 3 cimkéjüből pedig a Q változó értékét, azaz A = 25, P = 42, Q = 45.
- A 4 számu utasitás hatására a következő DATA elem ujra az 1 sorszámu utasitás első eleme
- Az 5 utasitás az S és T értékét olvassa be.

Igy S = 25, T = 42 utasitásnak megfelelő eredményt kapunk.

- A 6 utasitás hatására a következő DATA utasitás a sorrendben első DATA utasitás lesz, azaz most az 1 cimkéjü
- A 7 utasitás ennek megfelelően X-nek 25, Y-nak 42 értéket ad.

4.6.2. Kiiró utasitások

4,6.2.1. PRINT utasitás

Adatot a memóriából a képernyőre a PRINT utasitással irhatunk ki.

A PRINT utasitás formája:

$$\text{PRINT } l_1, l_2, \ldots, l_n$$

ahol l_i a kiirási lista elemei.

Lista elem lehet: egyszerü vagy indexes változó, karakter tipusu konstans, aritmetikai vagy logikai kifejezés.

Az utasitás a következőképpen hajtódik végre:

- Ha lista elem nem szerepel az utasitásban, akkor egy üres sor jelenik meg a képernyőn.

- Ha a lista elemei egyszerü vagy indexes változók, akkor minden elem kiirását az 1, 11, 21, 31 pozicióban kezdi. Igy egy sorban összesen 4 elemet tud kiirni a gép. Ha több, mint 4 elem van a listán, ugy uj sorban kezdődik a kiirás

- Ha a karakter tipusu konstans hosszabb, mint a fenti poziciók közötti elemek száma, ugy a következő "intervallum" üresen marad. Azaz, ha a lista elemei egy valós változó, egy 17 elemü karakter konstans és két egész változó, akkor egy sorba kerül a valós változó és a karakter konstans és az első egész változó és a következő sorba a második egész változó.
- Ha a listán egy aritmetikai kifejezés szerepel, a gép az értékét kiszámolja, majd kinyomtatja.
- Ha a lista eleme egy logikai kifejezés, akkor a gép meghatározza a logikai kifejezés értékét, majd a következőket irja ki:

0	ha az érték hamis
-1	egyébként

Megjegyezzük, hogy a lista elem elválasztó pontosvessző /;/ is lehet vessző/,/ helyett. Ennek hatására a következő történik:

- karakter tipusu változók /konstansok/ szünet nélkül egymás mellé kerülnek kiirásra
- numerikus változók értékei /valamint konstansok/ két szünettel lesznek elválasztva, ha pozitivok, egy szünettel és "-" előjellel, ha negativok. A pozitiv előjelet ugyanis nem irja ki a gép, helyette üres helyet hagy.

Ha a PRINT lista utolsó eleme után is irunk egy pontosvesszőt, akkor a következő kiiró utasitás listájának elemeit az előző után irja.

A PRINT kulcsszó teljes egészében helyettesithető a ? jellel.

4.6.2.2. Kiirás helyének vezérlése karakterekkel

PRINT utasitásban a kiirás helye vezérelhető utasitásokkal vagy a kurzor mozgatására szolgáló billentyükkel. Az erre vonatkozó utasitásokat a következő pontban ismertetjük. A kurzor mozgatására szolgáló gombokat PRINT utasitásban is elhelyezhetjük. Ez ugy hajtandó végre, hogy a kiirandó adat előtt idézőjelbe tesszük a megfelelő kurzor mozgatására szolgáló jeleket. Ezek a vezérlőjelek a következők:

billentyü	megjelenitve
a./ alsó állás, CRSR le	Q betü
b./ felső állás, CRSR fel	fekete kör
c./ alsó állás, CRSR jobb	szögletes zárójel
d./ felső állás, CRSR bal	függőleges vonal
e./ CLR	fekete sziv
f./ HOME	S
g./ INST	függőleges "bal vonal"

A vezérlőjelek hatására a következő adat kiirása az alábbiak szerint történik:

a./ Annyi sorral lejjebb történik a kiirás, ahány CRSR billentyü lenyomást végeztünk

b./ Annyi sorral feljebb történik a kiirás, ahányszor a CRSR billentyüt lenyomtuk

c./ Annyi hellyel jobbra történik a kiirás, ahányszor a billentyüt lenyomtuk

d./ Annyi hellyel balra történik a kiirás, ahányszor a billentyüt lenyomtuk

e./ A kiirás a bal felső sarokba történik, de a képernyő tartalma törlődik

f./ A kiirás a bal felső sarokba történik a képernyő tartalmának kitörlése nélkül

g./ Egy karakterhely "beszuródik" a kiirás elé.

4.6.2.3. Forditott szin és szines vezérlés

Ha a karakter szinét és az alapszint fel akarjuk cserélni a kiirás során, akkor a következőképp járunk el:

- lenyomjuk az idézőjel gombot
- lenyomjuk a CTRL gombot
- lenyomjuk a RVSON /9-es/ gombot
- lenyomjuk az idézőjel gombot.

Ezután a kiirandó elemek listája következik. A fenti elemeket a kiirandó adatok előtt, de az erre vonatkozó PRINT utasitásban kell elhelyezni.
A kiirandó elemek és alapszin visszacserélését a fenti módon lehet megtenni, de az RV$OFF /O nullás/ billentyüt kell lenyomnunk az RVSON helyett.
Az egyes kiirandó elemek szinét is megváltoztathatjuk az elem kiirására vonatkozó PRINT utasitásban.

A karakterszinek kiválasztásához a következő billentyüket használjuk:

Szin	Megjelenési forma
fekete	jobb oldali és felső vonal
fehér	E betü
piros	font jel
cinóber	alsó háromszög
lila	függőleges kockás vonal
zöld	felfele nyil /hatvány jel/
kék	balra nyil
sárga	"pi" jele

A fenti szinkiválasztást a következőképp kell használni

- PRINT utasitás begépelése
- idézőjel kezdet
- CTRL és a szin billentyü lenyomása
- idézőjel bezárva
- adatlista.

A fenti billentyüsorozat hatására a PRINT végrehajtása során az adatlista elemeit a kiválasztott szinben irja ki a gép.

Egyéb lehetőségek.

A PRINT utasitásban számos egyéb jel is megadható idézőjelek között. Ezek teljes skáláját szinte lehetetlen felsorolni. Néhányra itt kitérünk, amelyeket hamarosan tudunk alkalmazni. Igy például program futása közben áttérhetünk az alsó vagy felső állás beállitására /pl. kis és nagybetü/. Ezt a következőképp intézzük:

- Beütjük a PRINT utasitást, de idézőjelben helyet hagyunk a jelek későbbi begépelésének
- megnyomjuk a RETRURN gombot
- a lent megadott jelek közül a megfelelőt begépeljük a CTRL és az RVS ON billentyük lenyomása után.

Funkció	Tipus
alsó állás	N
felső állás	SHIFT N

4.6.2.4. Kiirás vezérlése SPC, TAB utasitásokkal

A kiirás helyét a SPC és TAB utasitásokkal is vezérelhetjük.

SPC utasitás

Az SPC utasitás formája a következő

SPC(I)

ahol I lehet konstans, változó, vagy aritmetikai kifejezés, amelynek értéke 0 és 255 között kell legyen. Hatására a következő változó I üres hely kiirása után került a képernyőre.

Példa

```
10 PRINT SPC(10);"AB";SPC(30);"ABC"
```

A futtatás után az AB szöveg 10 üres hely után kerül kinyomtatásra, az ABC szöveg pedig a következő sorban a 3.oszloptól kezdve.

TAB utasitás

Az utasitás formája a következő

TAB(I)

ahol I az SPC utasitásnál elmondott lehet.

Hatására a következő változó kiirása az aktuális sor I+1 oszlopában folytatódik.

Példa

```
10 PRINT TAB(10);"XY";TAB(20);"ZW"
```

Az XY szöveg kiirása a program végrehajtása során a 11., a ZW szöveg kiirása pedig a 21. oszlopban fog történni.

4.7. Vezérlő utasitások

Vezérlő utasitásoknak azokat nevezzük, amelyek a program utasitásainak végrehajtását megváltoztatják, vagy a program futását befejezik. Itt a következő utasitásokat fogjuk tárgyalni:

IF feltételes utasitás

GO TO ugró utasitás

FOR ciklus utasitás

STOP program logikai vége

END program fizikai vége

A program végrehajtási sorrendjét megváltoztató további utasitásokat /GOSUB, RETURN/ a szubrutinok tárgyalásánál mondjuk el.

4.7.1. IF feltételes utasitás

A feltételes utasitás a programozás egyik leggyakrabban használt utasitása. Akkor használjuk, ha a program végrehajtását valamilyen feltételtől függően el akarjuk ágaztatni.

Alakja a következő:

IF L THÉN u_1 :u_3:...:u_n

ahol L tetszésszerinti logikai kifejezés

u_i, /i=1,2,...,n/ tetszésszerinti végrehajtható utasitások.

Az utasitás végrehajtása a következőképp történik:

- ha az L kifejezés értéke igaz, akkor a gép az $u_1/u_2,...,u_n/$ utasitásokat hajtja végre;
- ellenkező esetben az IF utasitást követő sorszámu utasitás kerül végrehajtásra.

Az utasitás másik formája

IF L THEN i

ahol L megegyezik a fent elmondottal

i egy a programban található utasitás sorszáma.

Hatására:

- ha L igaz akkor az i-edik utasitásnál folytatódik a program végrehajtása;
- ellenkező esetben az IF utasitást követő sorszámu utasitás kerül végrehajtásra.

4.7.2. GO TO ugró utasitás

Sokszor szükséges a programban a müveletek végrehajtási sorrendjét megváltoztatni. Erre szolgál a feltétel nélküli ugró GO TO utasitás.

Formája a következő:

GO TO c

ahol c egy, a programban előforduló utasitás cimkéje.

Végrehajtása a következőképp történik: Ha a gép eljut egy GO TO utasitáshoz, akkor a végrehajtást a c cimkéjü utasitásnál fogja folytatni.

Példa:

```
10    A = 25
15   ?A
20    GO TO 300
30    A = 42
40   ?A
50    GO TO 500
300   A = 325
310   ?A
320   GO TO 30
500   .....
```

A programrészlet végrehajtása során az alábbi értékek kerülnek kinyomtatásra

25, 325, 42

Az utasitások végrehajtási sorrendje ugyanis a következő:

10, 15, 20, 300, 310

320, 30, 40, 50, 500

Létezik a GO TO utasitásnak egy másik alakja is BASIC-ben. Ennek végrehajtása egy változó értékétől függ. Az utasitást ezért feltételes ugró utasitásnak is szokás nevezni. Praktikusan használható olyan esetben, amikor a program végrehajtása egy sor irányban folytatódhat tovább, ágazhat el.

Ennek formája a következő:

ON v GO TO c_1, c_2, ... ,c_n

ahol v egész vagy valós tipusu aritmetikai kifejezés

c_i /i=1,2,...,n/ a programban előforduló cimke.

Az utasitás a következőképpen hajtódik végre:

1. Kiszámoljuk a v aritmetikai kifejezés értékét,
2. Ha az igy kiszámolt A értékre fennáll az $1 \leq A \leq n$ feltétel, akkor az utasitás ekvivalens egy

   GO TO [A]

   utasitással, ahol [A] az A változó egész értékét jelenti.
3. Ha 2. pontban leirt feltétel nem áll fent, akkor a futás a következő utasitásnál folytatódik, ha v értéke nem negativ. Negativ esetben a futás hibajelzést ad és a futás befejeződik.

Példa:

Tekintsük az alábbi programrészletet:

```
10  INPUT K
20  ON K+1 GO TO 30,40,50,60
30  ?K+1

40  ?K+2

50  ?K+3

60  ?K+4
```

A fenti programrészletben az aritmetikai kifejezés értékének egész része 1 és 4 között lehet.
Ezek után elkészithetjük a korábban tárgyalt képernyőt tisztára iró eljárást.
Ez a következő:

```
10  I = 0
20 IF I = 24 THEN 60
30 PRINT
40 I = I + 1
50 GO TO 20
60 .....
```

4.7.3. FOR ciklusutasitás

A fenti példában egy ciklust állitottunk elő. A 20-30-40 utasitások végrehajtását a gép ugyanis 25-ször végzi el.
A ciklus az alábbi felépitésü példánkban

a./ Ciklusváltozó /esetünkben I/ kezdeti értékének beállitása a 10-es utasitással

b./ Annak ellenőrzése, hogy a ciklusváltozó tulhaladta-e a végértéket, azaz I = 24 érték teljesülése.

c./ Ciklus belsejében - un. magjában levő utasitások végrehajtása - a fenti példában a 30-as utasitás.

d./ Ciklusváltozó értékének növelése - esetünkben a 40-es utasitás.

e./ Ugrás a ciklus elejére- esetünkben az 50-es utasitás.

A fenti eljárást összesen 25-ször hajtja végre a gép. Az I = 24 esetben beszünteti a ciklus magja végrehajtását, és a 60-as cimkéjü utasitásnál folytatja tovább a program végrehajtását.

A fenti programrészletet a FOR ciklusutasitással egyszerüen végrehajthatjuk.

A következő programrészlettel helyettesithetjük a fenti utasitássorozatot:

```
10 FOR I =0 TO 24
20 PRINT
30 NEXT I
 .
 .
 .
.........
```

A FOR-NEXT utasitáspár a következő alaku:

FOR v = m_1 TO m_2 STEP m_3

.....

.....

NEXT v

ahol v a ciklusváltozó /egész vagy valós tipusu/ nem indexes változó/,

m_1, m_2, m_3 aritmetikai kifejezések.

A ciklus utasitást a gép a következőképp hajtja végre:

1. Kiszámolja az m_1 kifejezés értékét és végrehajtja az v = m_1 értékadó utasitást.

2. Végrehajtja a FOR utasitást követő utasitásokat egészen a NEXT utasitás előtti utasitásig.
3. Növeli a ciklus változó értékét az m_3 kifejezés értékével, és visszaugrik a FOR utasitást követő utasitásra
4. Ellenőrzi, hogy v $<$ m_2 feltétel teljesül-e; ha igen 3-nál egyébként 5-nél folytatjuk a végrehajtást.
5. A következő végrehajtandó utasitást a NEXT v után következő.

Ha az m_3 kifejezés értéke 1, akkor a STEP m_3 részlet elmaradhat, azaz a következő két utasitás ugyan azt eredményezi:

```
FOR I = 1 TO 5 STEP 1

FOR I = 1 TO 5
```

A különböző gépi reprezentációkban a ciklusváltozó értékét rendszerint nem szabad megváltoztatni a ciklusson belül. Ebben a reprezentációban azonban ez a kikötés nem áll fenn, de ebben az esetben a ciklus nem a fent megadott számszor hajtódik végre.

Példa:

a./
```
10 FOR I = 1 TO 5
20 ?I
30 NEXT I
```

b./
```
10 FOR I = 1 TO 5
20 ?I
30 I = I + 1
40 NEXT I
```

Az a./ programrészlet eredményeképp az 1,2,3,4,5 számok kerülnek a képernyőre. A b./ esetben viszont 1,3,5 a 30-as I = I + 1 utasitásnak megfelelően. Igy az a./ ciklus ötször, a b./ csak háromszor kerül végrehajtásra. Az m_1, m_2, m_3 kifejezés értéke negativ is lehet. A végrehajtás ebben az esetben annyiban különbözik a fentitől, hogy a 2. pontban elmondott ellenőrzés módosul az alábbiakkal:

Ellenőrzi, hogy $v > m_2$ feltétel teljesül-e,

ha igen .....

Példa:

```
10 FOR I = 1 TO -5 STEP -1
20 ?I
30 NEXT I
```

A fenti példa esetén a ciklust összesen hétszer hajtja végre a gép.

Ennek megfelelően a kiirt értékek a következők: 1, 0, -1, -2, -3, -4, -5.

A programon belül megváltoztathatjuk az m_1, m_2, m_3 kifejezésekben szereplő változók értékét is. A ciklus ismétlési száma ebben az esetben sem tér el a fent mondottaktól, azaz ezeket a módositásokat a cikluson belül figyelmen kivül hagyjuk!

Ha a cikluson belül a v , m_1, m_2, m_3 értékeit nem változtatjuk meg, akkor a ciklus ismétlési száma a következő

$$\left[\frac{m_2 - m_1}{m_3}\right] + 1$$

ahol a [ ] jel az egész értéket jelöli.

Összefoglalva: a ciklus addig hajtódik végre, ameddig az alábbi kifejezés értéke igaz nem lesz:

$$m_3 < 0 \quad \text{AND} \quad v < m_2 \quad \text{OR} \quad m_3 > 0 \quad \text{AND} \quad v > m_2$$

Megjegyzés: az NEXT v helyett NEXT utasitást is használhatunk, azaz a változót nem kell kiirni az utasitás után. Ha több ciklust ágyazunk egymásba, akkor elegendő egy NEXT utasitást alkalmazni, amennyiben közös a befejezésük. Ekkor viszont az összes ciklusváltozót fel kell sorolni a NEXT után vesszővel elválasztva.

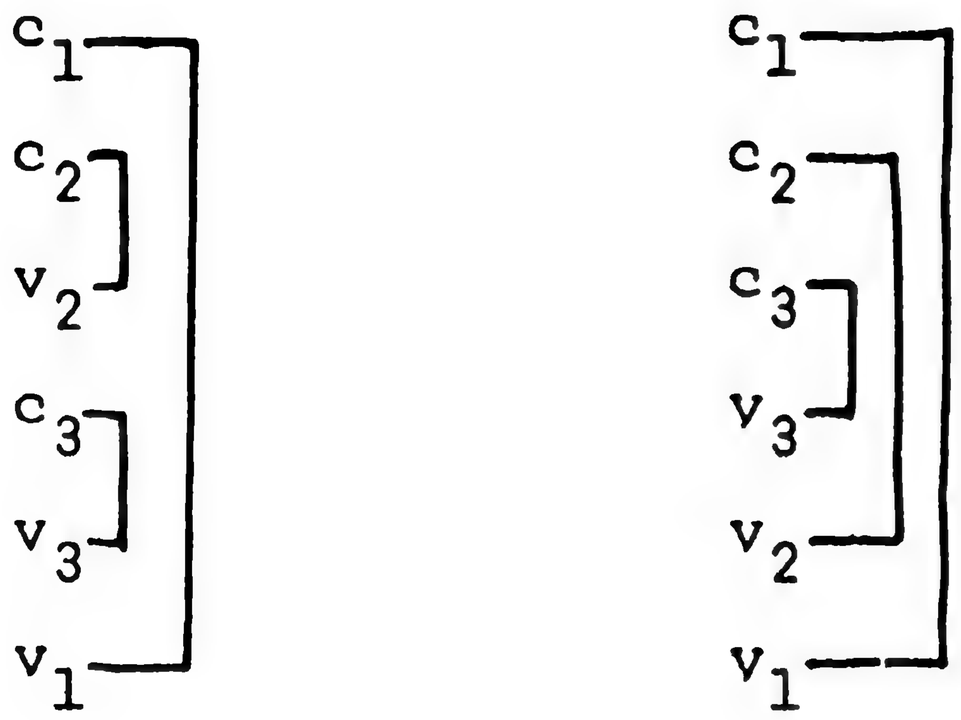
Példa:

Határozzuk meg az A tiz elemü vektor hosszát.

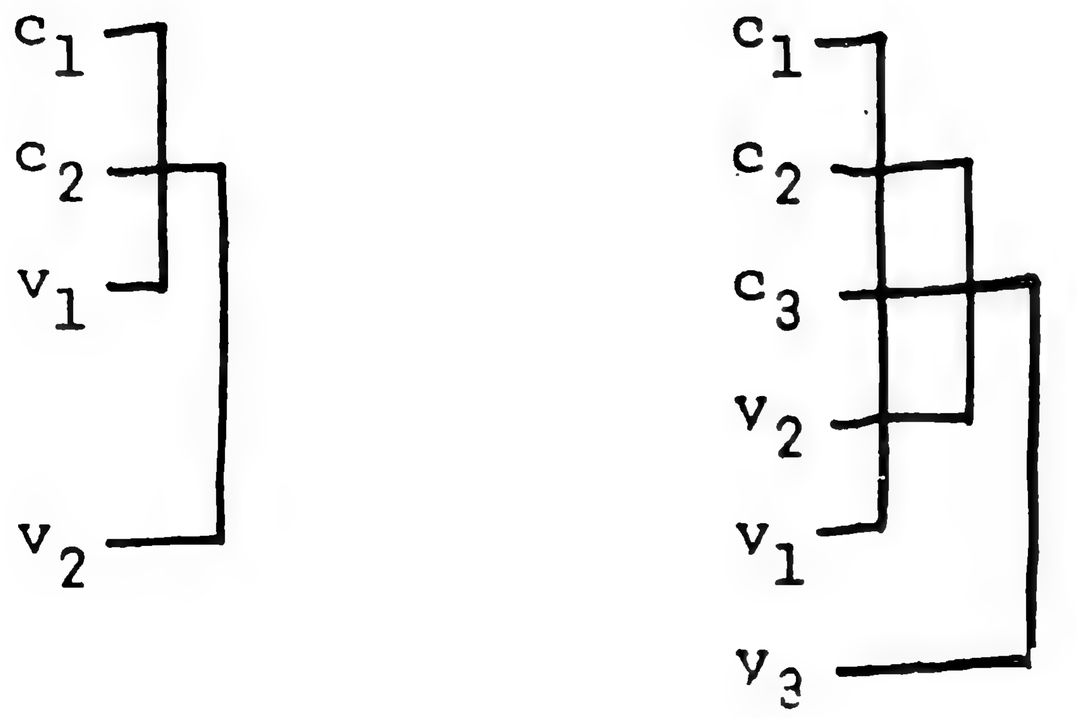
```
10 DIM A(10)
20 FOR I = 1 TO 10
30 INPUT A(I)
40 X = X + A(I)
50 NEXT I
60 Y = X ↑ 0.5
```

Több ciklust elhelyezhetünk egymáson belül is. Vigyázzunk azonban arra, hogy a külső ciklushoz tartozó NEXT utasitás minden belső ciklushoz tartozó NEXT utasitás után álljon.

Az egymásba ágyazott utasitások megengedettek, ha az alábbi formájuak:



és nem megengedettek, ha a következő alakuak:



ahol c_i a ciklus elejét, v_i pedig a ciklus végét jelöli.

Példák:

a./ Számoljuk ki az A és B mátrix összegét, és tegyük el az eredményt a C mátrixba.

```
10 DIM A(2,8), B(2,8), C(2,8)
20 FOR I = 1 TO 2
30 FOR J = 1 TO 8
40 C(I,J) = A(I,J) + B(I,J)
50 NEXT J
60 NEXT I
```

b./ Egymásba ágyazott ciklus végrehajtására tekintsük a következő ciklusokat:

```
10 FOR I = 1 TO 2
20 FOR J = 1 TO 3
30 FOR K = 1 TO 2
40 ;I,J,K
50 NEXT K
60 NEXT J
70 NEXT I
```

A programrészlet végrehajtása után a következő eredményeket irja ki a gép:

I	J	K
1	1	1
1	1	2
1	2	1
1	2	2
1	3	1
1	3	2

I	J	K
2	1	1
2	1	2
2	2	1
2	2	2
2	3	1
2	3	2

4.7.4. STOP és END utasitások

A STOP utasitás a program logikai végét jelzi. Mivel minden program több helyen befejeződhet, ezért a STOP utasitást több helyen is berakhatjuk a programba. A végrehajtás során a program kiirja annak a STOP utasitásnak a cimkéjét, amelyiknél a program befejeződött.

A STOP utasitás formája:

STOP

Üzenet a STOP hatására a programból:

BREAK IN LINE XXX

Az END utasitás hatása megegyezik a STOP utasitáséval.

Az END utasitás formája:

END

Az END utasitás végrehajtása után a gép a következőt irja ki:

READY

5. Programok kezelése

5.1. Futtatás, megállitás, ellenőrzés

A már gépben levő programot a kezelőpultról indithatjuk a RUN paranccsal.

Inditása: 1. RUN szó begépelése

2. RETURN billentyü lenyomása

Egy futó programot is megállithatunk. Erre ad lehetőséget a RUN/STOP billentyü.

Ennek megnyomása után a gép kiirja azon utasitás sorszámát, amelyet éppen végrehajt, azaz a

BREAK IN LINE XXX

üzenetet.

A futás során egy sor logikai hiba fordulhat elő. Ezek egy részét programból lekezelhetjük, más része viszont a program végrehajtásának azonnali befejezését jelenti.

A futás közben előforduló hibákat az I. függelékben foglaltuk össze.

5.2. Módositás, listázás, javitás, törlés

Program irás közben sokszor szükség van a program átnézésére, javitására, módositására. Erre különböző lehetőségeink vannak, ezeket vesszük sorra a továbbiakban.

Sor törlése a programból

Egy, már a gépben levő programsor kitörlése a következőképp történik:

- begépeljük a kitörlendő sor cimkéjét;
- megnyomjuk a RETURN billentyüt.

Egy sor beszurása a programba

Két utasitás közé egy harmadikat ugy szurunk be, hogy

- kiválasztunk egy olyan cimkét, amelyik értéke a két utasitás közé esik;
- begépeljük az utasitást a szokásos módon.

A gép az igy utólag beirt utasitást a cimke tényleges numerikus értékének megfelelő helyre teszi el. Amennyiben nincs lehetőség erre, mivel a két sor cimkéje egymásutáni numerikus értékek. akkor a programot átszámozzuk.

Program listázása

A már gépben levő programot a LIST paranccsal irathatjuk ki.

A LIST parancs az alábbi formákban használható:

a. LIST c

b. LIST c-

c. LIST -c

d. LIST c_1 - c_2

e. LIST

A LIST parancs begépelése után a RETURN billentyüt meg kell nyomni.

A gép a fenti parancsokat a következőképp hajtja végre:

a./ LIST c parancs hatására kiiródik a c cimkével ellátott utasitás;

b./ LIST c- parancs esetén a gép kilistázza a programot a c cimkével ellátott utasitástól kezdve;

c./ LIST -c parancs esetén a gép az első utasitástól kezdve a c cimkéjü utasitás után következő utasitásig kiirja a programot;

d./ LIST c_1-c_2 parancs esetén a gép a c_1 és c_2 cimkéjü utasitások közötti utasitásokat irja ki a c_1-et és c_2-t is beleértve;

e./ LIST parancs hatására kiirja a teljes programot.

Megjegyezzük, hogy a listázást lassithatjuk a CTRL gomb lenyomásával.

Program egy sorának módositása

Program egy sorát ugy módosithatjuk, hogy először a sort képernyőre listáztatjuk a LIST paranccsal, ha nem lenne olt.

Ezután a következőképp módositjuk az aktuális sort:

a./ a CRSR fel/le billentyüvel elmegyünk a kurzorral az illető sorhoz;

b./ a CRSR jobb/bal billentyüvel a javitandó elem utáni elemre állunk;

c./ ha törölni akarunk egy /több/ karaktert, akkor lenyomjuk az INST/DEL gombot egyszer /annyiszor, ahány elemet törölni akarunk/;

d./ ha <u>beszurni</u> akarunk arra a karakterre állunk rá, amelyik elé be akarunk szurni karaktert, vagy karaktereket;

- leütjük annyiszor a SHIFT, INST/DEL billentyüt, ahány elemet be akarunk szurni;
- begépeljük a szükséges elemeket az üres helyekre.

e./ mindkét esetben /törlésnél és beszurásnál/ elmegyünk az utasitás utolsó eleme után a CRSR jobb/bal billentyüvel és megnyomjuk a RETURN gombot.

<u>Program törlése a gépből</u>

Amennyiben uj programot irunk az esetleges gépben lévő programot ki kell törölni a gépből. Erre azért van szükség, mert ha az uj beirt program utasitásainak a száma nem egyezik meg a régi program utasitásszámaival, akkor a régi program utasitásai is bentmaradhatnak a gépben. A gép ugyanis ugy veszi, mintha a régi programot módositanánk az utasitással.

A program törlésére az alábbi módok vannak:

a./ Begépeljük a NEW parancsot, majd megnyomjuk a RETURN gombot

b./ Begépeljük a SYS 64738 utasitást és a RETURN gombot megnyomjuk. Ekkor a gép a törlés mellett alapállásba kerül, mintha most kapcsoltuk volna be

Mindkét parancs törli az aktuális programot és a hozzájuk tartozó adatterületet is.
Megjegyzendő, hogy a NEW parancsot betehetjük a programba is. Vigyázzunk azonban, mert a végrehajtás során nemcsak a futás fejeződik be, de a program is törlődik!

6. Program tárolása és betöltése háttértárolóból

A gépbe begépelt program a gép kikapcsolásakor vagy uj program gépbe vitelekor eltörlődik.
A többszöri felhasználáshoz ezért szükséges valamilyen a kikapcsolástól független háttértárolóra tenni /tölteni/ a programot.

6.1. Program elhelyezése háttértárolón

Program megőrzésére mikrogépeknél rendszerint mágnesszalagot vagy mágneslemezt használunk. Erre a mi esetünkben is lehetőség van.

Kiirás mágnesszalagra

Program mágnesszalagon való elhelyezésére a következő parancsot használjuk:

SAVE "fn"

ahol fn nem kötelezően használatos programnév. A programot szalagra a következőképpen helyezzük el.

a./ begépeljük a programot a gépbe;

b./ begépeljük a

SAVE "fn"

utasitást;

ha a magnetofon nincs bekapcsolva, akkor a gép

PRESS REKORD AND PLAY ON TAPE

üzenetet irja ki;

c./ betesszük a kazettát a magnetofonba;

d./ lenyomjuk a REC és PLAY gombot;

a gép OK

SAVING

választ küld és kiirja a programot

e./ a kiirás befejezése után kiirja a

READY

üzenetet.

Ezután a programunk a szalagon tárolásra került.

A kiirással kapcsolatban a következőket vegyük figyelembe:

- A programot szalagon mindig két példányban helyezi el a gép /igy betöltéskor korrigálhat/.
- A RECORD billentyü lenyomását nem ellenőrzi a gép, igy győződjünk meg ennek a lenyomásáról is. Ellenkező esetben a gép semmit sem vesz fel a magnetofonra.
- A szalag megfelelő helyéről gondoskodjunk. A befüző szalag fej előtti mozgását ugyanis normál szalagmozgásnak tekinti, igy erre is ráir. Az információk ez utóbbi esetben természetesen nem kerülnek kiirásra.

A kiirt program ellenőrzését végzi a VARIFY utasitás, amelyet a következő pontban ismertetünk.

Kiirás mágneslemezre

Mágneslemezre szintén irhatunk ki programot. Kiiratni szintén a SAVE utasitással lehet a következő formában:

```
SAVE "    ", 8
```

A kiirás a következőképpen történik:

a./ helyezzünk el egy megformázott lemezt a diszk egységbe;

b./ gépeljük be a

```
SAVE "xxxx", 8
```

utasitást;

c./ a gép válaszol

```
SAVING FOR xxxx
```

d./ a program kiirásának befejezésekor kiirja a

```
READY
```

üzenetet.

A SAVE utasitás általános alakja

```
SAVE "fn", d, 1
```

ahol "fn" a kiirandó program neve

d az egység fizikai sorszáma, amely mágnesszalag esetén 1, flopy diszk esetén 8; 1 jelenti, hogy a program kiirása után egy végjelet tesz a gép, amely a program betöltését segiti elő.

Példák:

SAVE	kiirja a gépben levő programot szalagra
SAVE "ALMA"	kiirja a gépben levő programot ALMA néven szalagra
SAVE "KLM",8	kiirja a gépben lévő programot lemezre KLM néven
SAVE A$,8	kiirja lemezre a gépben levő programot az A$ változó tartalma néven.

6.2. Program kivitel ellenőrzése

Amennyiben biztosak akarunk lenni a kiirás pontosságában, szükséges a programkivitelt ellenőrizni. Különösen ajánlható ez a kevésbé megbizható mágnesszalag esetén. A kazettára vagy lemezre kiirt programot a kivitelt követő READY üzenet után végezhetjük. Az ellenőrzést a gép ugy végzi, hogy a programot visszaolvassa a gépbe és összehasonlitja a gépben levő programmal. Ezért szalag esetén a szalagot vissza kell tekerni a program elejére. Az ellenőrző utasitás formája a következő:

VERIFY "f",n

ahol f a kiirt program neve

n az egység száma, amire a kiirás történt.

Szalag esetén a gép válaszol a VERIFY utasitásra:

PRESS PLAY ON TAPE

a PLAY gomb lenyomása után a gép kiirja:

VERIFYING

Az ellenőrzés eredményeképp két üzenet lehet.

a./ OK

azaz a kiirás hibátlan volt;

b./ ?VERIFY ERROR

azaz a kiirásban hiba volt, meg kell ismételni.

Megjegyezzük, hogy a VERIFY a szalagon az előző kiirt program végének megkeresését, azaz a következő üres hely megkeresését is segiti, mivel a hibaüzenet esetén ott áll meg a szalag.

6.3. Program betöltése háttértárolóból

A korábban háttértárolóra irt programot a LOAD utasitással tölthetjük be a memóriába.

Ennek formája a következő:

LOAD "fn" , d,c

ahol fn a betöltendő program neve

d a háttértároló egységszáma /1 szalag, 8 lemez esetén/

c parancs.

Betöltés szalagról a gépbe

A kazettás egységről a következő eljárással töltünk be egy programot:

a./ beütjük a LOAD "xxx" parancsot;

b./ ha a PLAY gomb nincs lenyomva, akkor a gép válaszol:

PRESS PLAY ON TAPE

a PLAY lenyomása esetén a c-nél folytatódik;

c./ gép válaszol /a szalagot elinditja/:

OK

SEARCHING

d./ ha az xxx nevü programot nem találja, akkor

?FILE NOT FOUND

szöveget irja ki és megáll;

e./ ha a program megvan, akkor a

FOUND

LOADING

szöveg jelenik meg a képernyőn;

f./ a betöltés után a

READY

szöveget irja ki a gép.

Ha nem vagyunk biztosak a program nevében, akkor csak a

LOAD

parancsot ütjük be.

Ekkor a gép a következő program megtalálásakor a

FOUND

programnév jelzést fogja kiirni.

Jegyezzük meg:

- Nem helyes programot és adatokat a gép nem tölt be szalagról!
- Ha a programot nem tudja betölteni a gép, akkor a következő üzenetet irja ki:

LOAD ERROR

Betöltés lemezről

A mágneslemezről a programot a következő paranccsal töltjük be:

LOAD "fn", 8

ahol fn a betöltendő program neve.

Az fn helyett használhatunk karaktertipusu változót is, amely a program nevét tartalmazza.

Ha a programot nem találja a gép

FILE NOT FOUND ERROR

üzenetet ir ki.

Ha a következő programot akarjuk betölteni lemezről, akkor a

LOAD "*", 8

programot használjuk.

Megjegyzés

1. A LOAD parancsot beirhatjuk a programba is.

   Ekkor a betöltés után a program végrehajtása azonnal elindul. Például:

   10 LOAD "ALMA", 8

   és RUN, RETURN

2. A betöltés során a gép hiba esetén a BASIC-ot is elronthatja. Ekkor kapcsoljuk ki, majd be a gépet.

6.4. Szabad memóriahely ellenőrzése

Programozás során sokszor szükséges a még rendelkezésre álló szabad memóriaterület ismerete.

Erre szolgál a FRE függvény, amely a következő alaku

FRE(x)

ahol x egy numerikus változó.

Hatására a FRE függvény a még szabad memória terület értékét veszi fel.

Példa

```
10 DIM A(60,100)
20 PRINT FRE(x)
RUN
```

Eredménye 8069

6.5. Irás/olvasás ellenőrzése

Az irás/olvasás ellenőrzésére van lehetőség a STATUS függvény segitségével. Formája a következő:

STATUS

A STATUS értéke numerikus érték. Minden irás/olvasás művelet után a STATUS felveszi a megfelelő értéket. Az irás/olvasás történhet kazettáról, képernyőről, billentyüzetről, soros bus-ról.

A STATUS értékeit a II. mellékletben levő táblázatban foglaljuk össze.

Példa:

```
10   OPEN 2,1,2, "AI."
20   GET # 2, A$
30   IF STATUS AND 64 THEN 60
40   ?A$
50   GO TO 20
60   ?A$ : CLOSE 2
```

7. Karaktersorozatokkal kapcsolatos müveletek

7.1. Értékadás

Mint korábban is elmondtuk, a karakteres változóknak is hasonlóan adhatunk értéket, mint az aritmetikai tipusnak.

Az értékadásra két lehetőség van:

- értékadó utasitással,
- kivülről beolvasással.

Értékadó utasitás az alábbi formátum lehet:

```
v = e
```

vagy

```
LET v = e
```

ahol v egy karakter tipusu egyszerü vagy indexes változó

e karakter konstans, változó vagy karakter értékü függvény.

Az értékadás ugy megy végbe, hogy a v változó balról jobbra az e értékét kapja.

Ha v hosszabb, mint e, akkor az értékadás megtörténik, és |v| - e számnyi betüköz /spaces/ kerül a v változó jobboldali részében /ahol |v| a v változó hosszát jelöli, lásd karakter tipusu változók/.

Ha v rövidebb, mint e, akkor a gép hibajelzést ad; a program végrehajtása ez esetben meg sem kezdődik. Az INPUT utasitást már tárgyaltuk. Ha ez karakter tipusu változókat olvas be, akkor a megadott konstans nem szükséges idézőjelbe tenni!

7.2. Karakter tipusu változók összeadása

Az összeadást értelemzhetjük karakter tipusu konstansok, változók esetén is. A müveletet a gép ugy hajtja végre, hogy a karaktereket az elemek sorrendjében egymás mellé irja.

Példa:

```
10 INPUT A$, B$
20 C$ = A$ + B$ + "IDE"
30 ;C$
```

A futás eredménye a különböző bemenő értékek esetén:

BE: A $	B $	KI: C $
ABC.	XQ	ABCXQIDE
15	21	1521IDE
TTTT	VTTV	TTTTVTTVIDE

7.3. Karakterekre vonatkozó függvények

A továbbiakban azokat a függvényeket ismertetjük, melyeknek argumentumai között karakter tipusu elemek /karakter tipusu konstansok, változók-függvények/ is lehetnek.

Ezen függvények értékei vagy karakter vagy numerikus tipusuak. Ennek megfelelően olyan kifejezésekben használhatjuk őket, amelyek tipusban összeegyeztethetők a függvény tipusával.

Az itt tárgyalandó függvények a következők:

Név	Tipus	M ü v e l e t
LEFT$	C	karakterek átmásolása
MID$	C	karakterek átmásolása
RIGHT$	C	karakterek átmásolása
LEN	VE	elem hossza
ASC	VE	elem első karakterének ASCI kódja
STR$	C	szám karakterré alakitása
VAL	V	numerikus karakterek numerikus számmá alakitása
CHR$	C	konvertálás ASCII kódból karakterré
POS	C	kurzor aktuális pozíciója

ahol a függvény tipusára használt jelek a következőt jelentik:

C karakter tipusu, V valós tipusu, E egész tipusu.

A továbbiakban programba beirható karaktersorozatokkal végzett müveleteket foglaljuk össze.

LEFT függvény

Formája a következő:

LEFT $ (k,r)

ahol k egy karakter tipusu konstans változó, vagy függvény

r egy egész vagy valós tipusu aritmetikai kifejezés

A függvény értéke a k első r eleme. Ha r valós tipusu volt, akkor először egésszé alakitja át.

Ha r értéke nulla vagy nagyobb mint a k változó hossza, akkor a futás befejeződik

Példa:

```
10 B $ = "ALMA A FA ALATT"
20 A $ = LEFT $ (B $, 5)
30 ?A$
40 C = 5
50 A $ = LEFT $ (B $, C * 2)
60 ?A$
70 A $ = LEFT$ ("ALMA A FA ALATT", 12)
80 ?A$
```

A fenti utasitások hatására a gép a következőket irja ki:

30. utasitás ALMA

60. utasitás ALMA A FA

80. utasitás ALMA A FA AL

MID függvény

Formája

MID$ (k,r_1,r_2)

ahol k karakter tipusu konstans változó vagy függvény,

r_1,r_2 egész vagy valós tipusu aritmetikai kifejezések.

A függvény értéke k r_1 pozíciójától kezdve r_2 számu elem. Ha r_1,r_2 valós tipusuak, ugy a végrehajtás előtt egésszé konvertálódnak.

Példa:

```
  5 C = 11
 10 B$ = "ALMA A FA ALATT"
 20 A$ = MID$ (B, 2, C + 1)
 30 ?A$
```

Eredmény: LMA A FA ALA

```
 10 A$ = MID$ ("ALMA A FA ALATT", 5, 10)
 20 ?A$
```

Eredmény: ,_, A FA ALAT

RIGHT függvény

Formája

RIGHT$ (k,r)

ahol k egy karakter tipusu elem /konstans, változó,
vagy függvény/

r aritmetikai kifejezés

A függvény értéke a k elem végétől kezdve a jobbra levő karakterek. Az r kifejezésre - valós esetén - a korábban elmondott szabályok érvényesek.

Példa:

```
10 INPUT N
20 A$ = RIGHT$ ("ALFABETA",N)
30 ?A$
```

Eredmény:

N értéke	kiirás
1	A
5	ABETA
9	ALFABETA
15	ALFABETA

LEN függvény

Formája:

LEN (k)

ahol k karakter tipusu elem /konstans, változó,
függvénykifejezés/

Értéke: a k kifejezés aktuális hossza.

Példa:

```
10 K$ = "ALMA A FA"
20 ?LEN (K$), LEN("HOL AZ OLLO")
```

Eredmény:

```
 9     11
```

ASC függvény

Formája:

```
ASC (k)
```

ahol k karakter tipusu elem /konstans, változo,
függvénykifejezés/

Értéke: a k kifejezés első karakterének ASCII kódja.
Az ASCII kódtáblázatot lásd a III. Mellékletben.

Példa:

```
10 A$ = "LENNI"
20 ?ASC (A$)
```

Eredmény: 76

STR$ függvény

Formája:

```
STR$(X)
```

ahol X egész vagy valós tipusu kifejezés.

Megadja a karakter reprezentációját az X értéknek.

VAL függvény

Formája:

VAL (k)

ahol k egy karakter tipusu elem.

Végrehajtása:

- ha k numerikus, akkor a függvény értéke az érték,
- ha k nem numerikus, akkor nulla az értéke.

Példa:

```
10 INPUT K$
20 ;VAL (K$)
```

Eredmény:

BE: K$	KI
1.2	1.2
2E3	2000
E	0

CHR függvény

Formája:

CHR$ (e)

ahol e aritmetikai kifejezés.

Értéke: az e aritmetikai kifejezések ASCII kódjának karakteres formája.

Példa:

```
10 INPUT A,B
20 ?CHR$ (A), CHR$ (2*B)
```

Eredmény:

BE: A, B	KI
76,41	LR
71,45	GZ
200,123	I+

A CHR függvény ezenkivül többféle szerepet lát el.
Az e értékétől függően más és más a végrehajtás.

POS függvény

Alakja a következő

POS(X)

A függvény értéke numerikus, mégpedig a kurzor aktuális oszlopának pozicióját adja meg. /Az első oszlop értéke nulla!/

8, Matematikai és egyéb függvények

8.1, Matematikai függvények

A programban egy sor matematikai kifejezés használható, amelyeket a BASIC tartalmaz.
Ezeket a függvényeket aritmetikai elemként lehet a programban alkalmazni, mint a konstansokat, változókat.
Ezen függvények aktivizálását függvényhivatkozásnak nevezzük. A függvények neve után zárójelbe téve egy aritmetikai kifejezést aktivizálhatjuk a függvényt.
Például sin/a + b/ + cos/c/ értéket az alábbi alakban számolhatjuk ki a BASIC-ben:

```
X = SIN(A + B) + COS (C)
```

Aritmetikai elemként a következő függvényeket használhatjuk:

SIN(X) kiszámitja az X aritmetikai kifejezés értékének szinuszát; X értékét radiánban kell megadni

COS(X) kiszámitja az X aritmetikai kifejezés értékének koszinuszát; X értékét radiánban kell megadni

TAN(X) kiszámitja az X aritmetikai kifejezés értékének tangensét; X értékét radiánban kell megadni

ATN(X) kiszámitja az X aritmetikai kifejezés értékének arkusztangensét, azaz azt a szöget, amelynek tangense X; a szöget radiánban szolgáltatja

LOG(X) kiszámitja az X aritmetikai kifejezés értékének természetes /e/ alapu logaritmusát; X > 0 értékek megengedettek

EXP(X) kiszámitja az e^X függvény értékét

SQR(X) kiszámitja X értékét; X $\geq$ 0 értékek megengedettek

INT(X) kiszámitja az X aritmetikai kifejezés értékének egész részét, vagyis azt a legnagyobb egészet, amely még nem nagyobb X-nél; például:

INT(3.27) = 3

INT(-8.15)= -9

ABS(X) kiszámolja az X aritmetikai kifejezés abszolut értékét

SGN(X) kiszámolja az X aritmetikai kifejezés értékére a signum függvény értékét

$$\text{signum/X/} = \begin{cases} -1 & \text{ha } X < 0 \\ 0 & \text{ha } X = 0 \\ 1 & \text{ha } X > 0 \end{cases}$$

A programba beirhatjuk a Ludolf féle számot is /3.14159/ aritmetikai elemként a következő formában:

SHIFT,↑

8,2. Véletlen szám generátor

A BASIC-ben véletlen szám generátor is be van épitve. A véletlen számot az RND generátor állitja elő, amely /0,1/ intervallumon szolgáltat egyenletes eloszlású számokat.

Alakja a következő

RND(X)

Az intervallum természetesen egyszerű szorzással módositható, Például a /0,10/ intervallumba a következőképpen állithatunk elő véletlen számokat:

X = RND(1)*10

Példa:

Töltsünk ki egy 10 x 10-es mátrixot 0 és 5 közötti véletlen számokkal.

A feladat az alábbi programrészlettel oldható meg:

```
20 FOR I = 1 TO 10
30 FOR J = 1 TO 10
40 A(I,J) = RND(1)*5
50 NEXT J
60 NEXT I
```

Az eloszlás ellenőrzésére irtuk a következő programot:

```
10  FOR V = 1 TO 10000
20  S% = RND(2)*10:D% = RND(1)*10
30  A(S%) = A(S%)+1:A(D%) = A(D%)+1
40  NEXT V
50  FOR I = 0 TO 9
60  PRINT A(I),B(I)
70  NEXT I
```

8.3. Időmérő függvények

A továbbiakban néhány függvényt fogunk megadni, amelyek hasznosak lesznek a gép müködtetéséhez.

Ezek a függvények a következők:

TI

TI$

A TI függvény a belső idő értéket veszi fel és század másodpercben adja vissza. Az érték kezdeti értékének beállitását érhetjük el a TI$ függvénnyel.

Példa:

```
PRINT TI/60
```

A TI$ függvény 6 karakterben az óra, perc, másodperc értékét adja. Be is lehet a kezdeti értéket állitani a megfelelő értékadó utasitással.

Példa

```
10  INPUT TI$:T$=TI$
20  IF TI$=T$ THEN 20
30  ?" ":TI$;:T$=TI$:GOTO 20
RUN
0010
```

A gép memóriájával kapcsolatos függvények

A gép memóriájának megváltoztatását és memória tartalom kiirását végző POKE és PEEK müveletekkel foglalkozunk a továbbiakban.

A POKE utasitás a következő formátumu:

```
POKE h,e
```

ahol h egy memóriarekesz cime 1-től 65536-ig

értékeket vehet fel

e egy tárolható érték.

A h cim nem lehet a teljes memóriatartomány. Azokat a cimeket, amelyek értékét megváltoztathatjuk, a memória térképből tudhatjuk meg. A POKE utasitás hatására a h-val megadott memória cimre az e érték kerül tárolásra.

Ezt az utasitást elsősorban a grafikus megjelenitésnél, szinek változtatásánál, zenei effektusok létrehozásánál alkalmazzuk, Kiadható mind paraméterként, mind programban utasitásként. Mivel minden egyes memóriaelem egy byte hosszuságu, ezért e értéke 0 és 255 között lehet. A PEEK függvénnyel egy megadott memóriarekesz tartalmát irathatjuk ki. Alakja a következő

PEEK(I)

ahol I 0 és 65535 közötti érték.
Hatására a gép kiirja az I. memória rekeszben levő értéket, azaz egy 0 és 255 közötti értéket.

8.4. Saját függvény megadása

Az előző fejezetben láttuk, hogy egyszerü matematikai függvényeket használhatunk aritmetikai elemként a programba. Lehetőség van saját függvény megadására is a BASIC nyelvben.

A függvény megadását a következő utasitással érjük el:

DEF FNXX(Y) = e

ahol XX a függvény neve, egy egész vagy valós változó neve

Y változó a függvény un. formális paramétere

e aritmetikai kifejezés.

A függvény értékének kiszámitása előtt az Y változó helyébe be kell irni azt a V un. aktuális paramétert, amely esetén ki akarjuk számolni az e kifejezés értékét.

A V aktuális paraméter lehet:

konstans

egyszerü vagy indexes változó

aritmetikai kifejezés.

Példa:

```
10 DEF F NQ(X1) = X1 + 1/X1*(X1+12)+5
```

A fenti függvény értékeit számoljuk ki 6, 7 értékekre.

Ezt a következő utasitások végzik el:

```
100 V = FNQ(6):S = FNQ(7)
```

A fenti példában a formális paraméter: X1, az aktuális paraméterek 6,7. Az utasitás végrehajtása a következőképpen történik:

1, A gép kiszámolja az aktuális paraméterlistán szereplő aritmetikai kifejezés értékét.

2. Elvégzi az Y = V értékadó utasitást.

3. A V értékre elvégzi a kijelölt müveletet.

4. Ezt az értéket rendeli hozzá az FNXX-hez.

Igy a fenti példa esetén elvégzi az X1 = 6 értékadó utasitást. Kiszámolja az aritmetikai kifejezés értékét, ez 14, és elvégzi V = 14 értékadó utasitást /a 100-as utasitás miatt!/. Majd X1 = 7-re végrehajtja ugyanezt.

9, Szubrutinok készitése

A BASIC-ben is készithetünk szubrutint, mégpedig ennek egy primitivebb változatát az un. nyilt szubrutint. A szubrutin bármely utasitással kezdődhet és egy RETURN utasitással fejeződik be.

A RETURN utasitás alakja:

RETURN.

tehát semmiféle paramétert nem tartalmaz.

Szubrutint a GOSUB utasitással hajthatunk végre.

Az utasitás alakja:

GOSUB c

ahol c utasitáscimke, a szubrutin első végrehajtandó utasitásának cimkéje.

A GOSUB utasitás a következőképpen ahjtódik végre:

1. A gép megjegyzi a GOSUB utasitás után követő utasitás helyét /legyen ez c_1/.
2. A program végrehajtása a c utasitásnál folytatódik, és egészen addig tart, ameddig el nem érjük az első RETURN utasitást.

3. A program végrehajtása a c_1 cimkéjü utasitással folytatódik tovább.

Példa:

Készitsünk egy szubrutint, amely a képernyő tartalmát kinullázza, és használjuk a 100-as és 200-as utasitás esetén ezt.

```
10.........

...........

100 GOSUB 500

110 X = 52

............

.............

200 GOSUB 500

210 Y = X + 5

.............

.............

     STOP

500 FOR I = 0 TO 24

510 PRINT

520 NEXT I

530 RETURN
```

A program végrehajtása a következő lépésekben történik:

10, 20..., 100, 500-530, 110...

...200, 500-530, 210....

Ha egy szubrutint több adatsorra is végre akarunk hajtani, akkor nekünk kell gondoskodni a megfelelő értékek betöltéséről, mivel a gép csak nyilt szubrutint tud kezelni.

Lehetőség van arra, hogy valamilyen értéktől függően más és más szubrutint hajtsunk végre.

Ezt a következő utasitás végzi el:

ON v GOSUB $c_1,c_2,\ldots,c_n$

ahol v egy aritmetikai kifejezés

c_i /i=1,2,...,n/ cimkék.

Az utasitás a következőképpen hajtódik végre:

1. A gép kiértékeli a v aritmetikai kifejezés értékét
2. Ha $1 \leq v \leq n$, akkor az utasitás ekvivalens egy

GOSUB [v]

utasitással, ahol [ ] az egész érték függvényt jelöli.

3. Ha a fenti feltétel nem teljesül, akkor a gép az ON.... utasitást követő utasitást hajtja végre, ha v értéke nem negativ. Ha v értéke negativ, a gép hibaüzenettel befejezi a futást.

Példa:

Tekintsük az alábbi programrészletet:

```
 10  INPUT K
 20  ON K+1 GOSUB 100,200,300
 30  G...

100  ?"ELSO"
110  RETURN
200  ?"MASODIK"
210  RETURN
300  ?"HARMADIK"
310  RETURN
```

A program végrehajtása a K értékétől függően a következőképpen történik:

K értéke	Kiirás
0	ELSO
2	HARMADIK
4	futás befejeződik
-1	futás befejeződik.

10. Háttértárak használata adattárolásra

A továbbiakban vázoljuk a C64 gép file kezelő rendszerének használatát. Mivel az alapgéphez sokszor csak kazettás magnetofont vesznek meg, ezért az utasitások bemutatása elsősorban ezen az egységen történik.

Megjegyzendő, hogy az alapvető különbség a mágnesszalag és a floppy egység között a következő:

A. Mágnesszalag esetén az adatokat csak olyan sorrendben olvashatjuk vissza, amilyen sorrendben kiirtuk. Egy elem megkeresése ezért sokáig tart: addig olvasni a szalagot, ameddig az elemet meg nem találtuk /un. soros állománykezelés!/.

B. Floppy egység esetén lehetséges olyan feldolgozás is, hogy a kivitelkor egy-egy rögzitett helyre történik az elemek kiirása, a visszaolvasás pedig tetszésszerinti sorrendben /direkt állománykezelés/.

Ilyen feldolgozásnál egy-egy elemet a helyének megadásával azonnal beolvashatunk a lemezről.

A fenti megjegyzésből következik, hogy feldolgozásokhoz célszerübb a floppy egység használatát igénybevenni. Ez az egység a gyorsasága mellett sokkal megbizhatóbb is.

10.1. Kazettás magnetofon használata

Kazettás egységeken történő adattárolás során az alábbi utasitásokat használjuk:

OPEN	adatállomány megnyitása
CLOSE	adatállomány lezárása
PRINT	adatkiirás mágnesszalagra
INPUT	beolvasás mágnesszalagról
GET	egy jel beolvasása szalagról.

Megjegyezzük, hogy mágneslemez esetén ugyanezeket az utasitásokat használjuk, csak ezek az utasitások más paramétereket tartalmaznak.

A továbbiakban végigvesszük a fenti utasitásokat.

10.1.1. Adatállomány létrehozása

Az adatállomány létrehozása három fázisból áll:

a./ Adatállomány elnevezése és az azonositó
felirása mágnesszalagra /un. file kezdő cimke/

b./ Adatok kivitele

c./ Adatállomány végének jelölése.

Az adatállomány azonositó cimkéjének felirását az OPEN utasitás végzi.

Az utasitás formája a következő:

```
OPEN  fa, l, v, "fn"
```

ahol fa az adatállomány azonositója a programon belül, értéke 1 és 255 közötti egész szám,

v parancs

fn az adatállomány neve, ami a mágnesszalagra is kiirásra kerül.

A v parancs értékei a következők lehetnek:

1 akkor irás történik a kazettára

2 akkor irás történhet, és az állomány végére végcimke is kiirásra kerül /EOT/

Példák:

```
OPEN    25, 1, 1, "ALMA"
OPEN     2, 1, 2, "ADATALLOMANY"
```

Az első utasitásban az adatállomány azonositója 25, az állományt irásra nyitottuk meg, a szalagra kiirandó név ALMA.

A második esetben az adatállomány programon belüli azonositója 2, az állomány végére egy állomány vége jel is kiirásra kerül, és az adatállomány neve ADATALLOMANY lesz. Az OPEN paranccsal kapcsolatban a következőkre vigyázzunk:

- a mágnesszalag az OPEN végrehajtása előtt a magnetofonban legyen;
- a fej előtt mágnesezhető rész legyen, ne befüző szalagrész.

A gép az első utasitást a következőképpen hajtja végre:

- kiirja a

PRESS RECORD PLAY ON TAPE

üzenetet;

- a két billentyüt lenyomva á képernyő képe eltünik, és a szalagot a gép meginditja;
- körülbelül 15 mp mulva a kép ujra megjelenik és a gép a READY üzenetet irja ki.

Megjegyzendő, hogy ha véletlenül csak a PLAY billentyüt nyomjuk meg /vagy esetleg más billentyüt/, az egész folyamat lejátszódik de kiirás természetesen nem történik. Amennyiben a már kiirt szalagot be akarjuk olvastatni, akkor a szalagot olvasásra kell megnyitni. Az olvasáshoz tartozó v parancs értéke 0, azaz az utasitás a következőképpen néz ki:

OPEN fa, 1, 0, "fn"

ahol fa ugyanaz mint irásra megnyitáskor

fn adatállomány neve.

Az adatállomány nevének meg kell egyezni a kiiráskor megadott adatállomány névvel.
Ezután az adatállomány kiirásra meg van nyitva és következhet az adatkiirás.

10.1.2. Adatkivitel kazettára

Az adatok kiirása a képernyőhöz hasonló módon a PRINT, illetve az ezzel ekvivalens; utasitás valamelyikével történik.

A kiirás előtt gondoskodni kell az adatállomány megnyitásáról, amit az OPEN utasitással érünk el.
Az utasitás formája a következő:

PRINT #n , l

ahol n az OPEN utasitásban megadott adatállomány azonositó,
l a kiirandó adatok listája.

Az utasitás hatására az l lista elemei kiiródnak az n-nel azonositott adatállományra.

Példa:

Tekintsük az alábbi egyszerü feladatot: irassuk ki az 1, 2, 3, 4 ,...., 10 értékeket mágnesszalagra.
A feladatot az alábbi programrészlet végzi el:

```
10  OPEN 21, 1, 1, "ALMA"
20  FOR I = 1 TO 10
30  PRINT #21, I
40  NEXT I
50 CLOSE 21
```

A fenti programrészlet a következőket végzi:

- 10-es utasitás "elkészit" egy ALMA nevü file-t, amelyre itt a programon belül a 21 azonositóval fogunk hivatkozni.
- a 20-40 ciklusban a 30-as utasitás a 21-es file-re kiirja I értéket.

Igy a szalagra az 1, 2, ... ,10 értékek kerülnek kiiratásra 10 egymásutáni alkalommal.

A CLOSE utasitásról a következő paragrafusban beszélünk.

A mágnesszalag kihasználása ebben az esetben nem megfelelő, mivel tiz "rekord" kerül kiirásra, a 20 utasitást összesen tizszer hajtjuk vére.

Ennek kiküszöbölésére lehetőség van arra, hogy egyszerre több adatot is kivigyünk szalagra, azaz, hogy egy rekord több adatot is tartalmazzon. Ekkor vigyázni kell arra, hogy a beolvasó utasitás számára megfelelő formátumu adatokat irjunk ki. Az egyes adatok közé tehát "," kell tenni, hogy később a beolvasó utasitás egymástól el tudja választani őket.

Példa:

Készitsünk programot, amely 50 rekordot ir ki összesen és egy rekordba négy elem kerüljön. A kiirandó rekordokat a programban egy külön ciklusban állitjuk elő az alábbi módon:

```
10  OPEN 12, 1, 1, "ASDF"
20  FOR I = 1 TO 50
30  FOR J = 1 TO 4
40  B/J/ = I + J
50  NEXT J
60  PRINT #12, B(1), ",", B(2), ",", B(3), ",", B(4),";"
70  NEXT I
80  CLOSE 12
90  END
```

A programban az I ciklus összesen ötvenszer fut le és a 12-es azonositóju adatállományba kiirja az aktuális B/1/, B/2/, B/3/, B/4/ értékeket. Az adatok megfelelő formára való hozását a 60-as utasitásban végezzük el. Az adatok közé karakter konstansként kiirjuk az adatokat elválasztó "," jelet. Ennek megfelelően az adatok az alábbi alakban iródnak ki a mágnesszalagra:

1. rekord	2, 3, 4, 5
2. rekord	3, 4, 5, 6
3. rekord	4, 5, 6, 7
.	
.	
.	
40.rekord	50, 51, 52, 53
50. rekord	51, 52, 53, 54

A kezdő cimke, az adatok és a végcimke felirására összesen 31 fordulat volt szükséges, a kiiráshoz szükséges idő pedig 2 perc 10 másodperc volt.

Megjegyezzük, hogy az elválasztójelet másképpen is beirhatjuk az elemek közé. Erre mutatunk példát az alábbiakban, ahol 10 rekordot irunk ki, amelyek mindegyike 3 karakter tipusu adatot tartalmaz.

```
10  OPEN 1, 1, 1, "SZALAG"
20  R$ = CHR$(13)
30  FOR I = 1 TO 10
40  INPUT A$, B$, C$
50  PRINT #1, A$, R$, B$, R$, C$, R$
60  NEXT I
```

A CHR$/13/ a RETURN billentyü megnyomásának felel meg. Az eddig bemutatott"," és CHR$(13) jelek mellett természetesen";" elválasztójelet is alkalmazhatunk.

10,1.3. Adatállomány lezárása

Ahhoz, hogy az adatállomány normálisan befejeződjön, az utolsó adat kivitele, vagy kiolvasása után le kell zárni, Ezt a CLOSE utasitás végzi.

A CLOSE utasitás alakja a következő:

CLOSE n

ahol n az adatállomány száma,amellyel megnyitottuk.

10.1.4. Beolvasás kazettáról

Már mágnesszalagra kiirt adatokat az INPUT utasitással olvashatunk be a gép memóriájába. Az első olvasás előtt gondoskodni kell az állomány megnyitásáról, azaz egy OPEN utasitás kiadásáról. Mint az OPEN-nél elmondtuk, a magnetofont be kell kapcsolni, gondoskodni kell, hogy a megfelelő helyre beállitsuk az aktuális kazettát és benyomjuk a PLAY billentyüt.

Az INPUT utasitás alakja a következő:

INPUT #n,l

ahol n az OPEN utasitásban megadott adatállomány azonositója

l a beolvasandó adatok listája.

A listánál ügyelni kell arra, hogy:

- a lista nem lehet hosszabb, mint az állományba kiirt adatok listája;
- a lista elemeinek tipusban rendre meg kell egyezni a kiiráskor figyelembe vett tipussal.

Példák:

Az alábbiakban közöljük a kiirásnál megadott 50 rekordot tartalmazó szalag beolvasását végző programrészleteket.

A program eredményének beolvasását az alábbi programrészlet végzi:

```
100 OPEN   12, 1, 0, "ASDF"
200 FOR I = 1 TO 50
300 INPUT #12, B(1), B(2), B(3), B(4)
400 PRINT B(1), B(2), B(3), B(4)
500 FOR J = 1 TO 1000:NEXT J
600 NEXT I
700 CLOSE 12
```

A program végrehajtásának eredménye megjelenik a képernyőn is:

```
2    3    4    5
3    4    5    6
4    5    6    7
. . . . . .
```

10.2. Egyéb perifériák kezelése

10.2.1. Perifériák megnyitása

Mint a szalagnál elmondtuk, a használat előtt a megfelelő egységet meg kell nyitni. A megnyitást az OPEN utasitással végezzük el, amit kiadhatunk parancsként - programon kivül - és utasitásként programon belül.

Az OPEN utasitás formája a következő:

OPEN fn, dv, c, s

ahol fn az eszköz azonositója, értékei 1 és 255 között lehetnek; ezt fogjuk az INPUT, GET, PRINT utasitásban használni, ha ezzel az eszközzel dolgozu

dv az egység számot jelöli, amit az alábbi táblázatban adunk meg

c parancs, amely az eszköz használatának tipusára vonatkozik, értékét a későbbiekben közöljük

s string tipusu érték.

Mint az alábbi táblázatból is látszik, ezt a parancsot használhatjuk a képernyő, kazettás magnó, nyomtató vágy diskegység esetén is. Ha megnyitjuk az adatcsatornát egy ilyen külső egységnél, akkor a gép egy puffer területet rendel az egységhez és egyszerre egy-egy puffer területet visz át a gépből/gépbe.

A dv egységtipus a következő lehet:

dv	egység
0	billentyüzet
1	kazettás egység
2	RS232-re kapcsolt egység
3	képernyő
4	nyomtató
5	nyomtató
8	disk egység
4 - 127	soros busz-on levő egységek száma

A c parancs értékei a következők lehetnek.

Egység	c	hatása
kazetta	0	kazetta olvasás
kazetta	1	kazetta irás
kazetta	2	kazetta irás, végcimke irása az irás befejezése után
disk	1 - 14	adatcsatorna megnyitása
disk	15	parancs csatorna megnyitása
billentyüzet	1 - 255	nincs hatással
képernyő	1 - 255	nincs hatással
nyomtató	0	felső állás/grafika
nyomtató	7	felső/alsó állás
RS232		egy sor parancsot kiadhatunk, ezek felsorolása az egységet leiró kézikönyvekben találhatók

Az s értéke egységtől függő. Hatása nyomtató vagy képernyő esetén megegyezik egy PRINT s utasitással.
Ha kazettás egységről van szó, akkor s az adatállomány azonositója, vagy esetleg egy további parancs.

10.2.2. Irás/olvasás

Az egyes egységekre ugyanugy lehet felirni adatokat, amint azt a kazettás egységnél elmondtuk. Formája is megegyezik a PRINT utasitásnál korábban elmondottakkal.

Formája a következő:

PRINT #fn, l

ahol fn az OPEN-ben megadott egység azonositója

l a kiirandó adatok listája.

Az l lista elemeit ugyanugy irja ki, mint a PRINT utasitásban, azaz nincs elválasztó jel az adatok elemei között. Ezért az elválasztó jeleket be kell irni a PRINT lista elemei közé. Ezt megtehetjük karakter konstansként /","/ illetve a CHR$/13/ elemek közé való irásával. Például:

PRINT #1, A; CHR$(13); B; CHR$(13); C

Adatállományba kiirt adatot az INPUT utasitással lehet szalagról, lemezről leolvasni.

Ennek formája a következő

INPUT #fn, l

ahol fn az OPEN-ben megadott egység azonositó száma

l beolvasandó adatok listája.

Az utasitás egy-egy változóba egy adatot hoz be - egészen a kiirt elválasztó jelig.

A GET utasitással egy byte-t lehet beolvasni külső egységről.

Formája a következő:

GET #fn, v

ahol fn megegyezik az INPUT-nál elmondottakkal,

v változónév.

Ha nincs az egységre adat kiirva,akkor 0 érték kerül a v változóba. Az utasitással tetszésszerinti karaktert beolvashatunk beleértve az idézőjelet és a kocsi visszasoremelést /CR/.

10.2.3. Egységek lezárása

Az egység lezárását ugyanugy végezzük, ahogy a kazettás egységnél elmondtuk:

CLOSE fn

ahol fn a lezárandó egység OPEN-ben megadott azonositója.

10,3, Mágneslemez használata

A COMMODORE géphez többfajta mágneslemez hozzákapcsolható. A legegyszerübb és közvetlenül hozzácsatlakoztatható egység a VC 1541-es tipusszámot viseli. A lemezt alapvetően két funkcióra használjuk:

a./ Programtárolás

b./ Adattárolás

A lemezre irni, onnan olvasni csak megfelelő előkészités, formázás után lehet. A géphez adott rendszerlemezen egy sor program található, amelyek között van a formázóprogram is. A továbbiakban elmondjuk a fontosabb lemezmüveleteket és kitérünk a lemezhasználat elemeire is.

10.3.1. Rendszerlemez programjai

A rendszerlemez minden előzetes formázás nélkül használható a gépen. A lemezen egy sor a munkát segitő program helyezkedik el.

A lemezen levő programok a következők:

1541 TEST/DEMO

HOW TO USE

HOW PART TWO

VIC-20 WEDGE

C-64 WEDGE

DOS 5.1

COPY/ALL

PRINTER TEST

DISK ADDR CHANGE

DIR

VIEW BAM

CHECK DISK

DISPLAY T & S

Egy-egy program listáját kiirathatjuk a következő utasitással:

LOAD "pr",8

ahol pr a fenti programnevek valamelyike

8 a floppy egység fizikai száma.

Megjegyezzük, hogy bizonyos programok le vannak védve a kiiratásra, igy sem printerre, sem képernyőre nem irathatók ki.

A fenti rendszerprogramok közül a fontosabbakat irjuk le a továbbiakban.

HOW TO USE

Az angol leirását adja a rendszerprogramnak és néhány egyéb programnak. A HOW PAR TWO a fenti leirás folytatása.

COPY/ALL

A disken elhelyezkedő adatok /programok/ másolására szolgáló rendszerprogram. A programmal csak két lemezes rendszerrel másolhatunk, ezért az egyik lemezt át kell nevezni a DISK ADDR CHANGE programmal.

PRINTER TEST

A fenti program a sornyomtatót teszteli le az összes karakter kiirásával. Többek között ellenőrzi a printer mechanikus és elektronikus részét.

VIEW BAM

A szektorok tartalmának ellenőrzésére szolgáló rendszerprogram.

DIR

A program kilistázza a lemezen elhelyezkedő programok, adatállományok nevét.

CHECK DISK

Egy lemezt ellenőriz le abból a szempontból, hogy a lemezen nincs egy hibás blokk. A program végigir minden egyes blokkot és a hibásakat megjegyzi a BAM-ban.

DISPLAY T & S

Kiirja a trekkek és szektorok tartalmát megadott trek és szektorszámok esetén.

PERFORMANCE TEST

A program a lemezegység ellenőrzésére használható /mechanikus és elektronikus ellenőrzés/.

SEQUENTIAL FILE

Példa soros adatállomány létrehozására.

RANDOM FILE

Példa véletlenszerü állomány létrehozására.

A lemezt használhatjuk a gépbe már beütött program elrakására a következőképpen:

```
SAVE "fn", 8
RETURN
```

ahol fn a program neve

8 a lemez fizikai egységszáma.

Megjegyezzük, hogy a parancs kiadása előtt gondoskodnunk kell a megformázott lemez gépbe tételéről, a gép bekapcsolásáról.

Adatot szekvenciálisan lényegében a magnetofonnál elmondott módon irathatunk mágneslemezre. Direkt kiirásnál először egy pufferterületre kerülnek az adatok, majd innen iródnak ki a lemezre. A használathoz segitséget ad a fennt emlitett program és a lemezre vonatkozó kézikönyv. A kétfajta állománykezeléssel a következőkben foglalkozunk.

10.3.2. Soros állománykezelés

Soros állománykezelés esetén az összetartozó és egyszerre kiirásra /beolvasásra/ kerülő adatok csoportját /un. rekordokat/ egymásutáni mágneslemez területekre helyezzük el. Ennek megfelelően a visszaolvasás is ilyen sorrendben mehet csak végbe.

A feldolgozás előnye, hogy helykihagyás a lemezen nem történik. Hátránya viszont, hogy a feldolgozás során egy elem megkeresése csak a kiirás sorrendjében történő olvasás során történhet. Ezenkivül egy-egy rekord tartalmának megváltoztatását sem érhetjük el, csak az egész állomány átmásolása utján. Uj rekordokkal viszont bővithetjük az állományt, mégpedig a rekord állomány végére történő irásával. A soros állomány kezelésére mutatunk be egy példát, amelyben két numerikus és egy karakter tipusu változóból álló adatsorokat /rekordokat/ irunk ki egymásután.

Soros állomány létrehozásának lépései:

1, Állomány megnyitása /hasonlóan a mágnesszalagnál elmondottakhoz/:

```
OPEN  2, 8, 2, "0: ADAT, S, W"
```

Az utasitás hatására a 8 egységen /lemez!/ olvasásra megnyitottuk az ADAT nevü állományt irásra

2. Adatok kiirása sorban a következő utasitással

/X$ = CHR$/13//

```
PRINT #2,A; X$; B; X$; C$; X$;
```

3, Állomány lezárása:

```
CLOSE 2
```

A felvitelt az alábbi programrésszel készitjük el:

```
10  OPEN 2, 3, 2, "0:ADAT, S, W" : X$ = CHR$/13/
20  FOR I = 1 TO 1000
30  INPUT A, B, C$
40  PRINT #2, A; X$; B; X$; C$; X$;
50  NEXT
60  CLOSE 2
```

A fenti programrészlet összesen 1000 adathármast ir ki lemezre /A, B, C$/. Az X$ változó az adatok közé tesz elválasztójelet /lásd kazettás magnetofon állománykezelés/. Az adatok visszaolvasásához a következő müveletek elvégzése szükséges:

4. Állomány megnyitása olvasásra:

```
OPEN 2, 8, 2, "0:ADAT, S, R"
```

5. A kivitt adatok egymásutáni beolvasása a következő utasitással:

```
INPUT #2, A, B, C$
```

6. Az állomány lezárása

```
CLOSE 2
```

A fenti 1000 számhármast a következő programmal olvastatjuk be:

```
 70  OPEN 2, 8, 2, "0:ADAT, S, R"
 80  FOR I = 1 TO 1000
 90  PRINT#2 , A, B, C$
100  PRINT A, B, C$
110  NEXT
120  CLOSE 2
```

10.3.3. Direkt állomány

Direkt állománykezelés esetén a gép és a floppy lemez mélyebb ismerete szükséges. A részletes állomány-kezelés ismertetését az egység kezelését leiró kézikönyv részletesen tartalmazza.

Itt csak az állománykezelés alapjait ismer-tetjük.

A floppy lemez összesen 664 255 byte hosszuságu fizikai rekordból áll. A fizikai rekordok /sektor/ koncentrikus körökön, track-keken helyezkednek el. A külső körökön több, a belsőkön kevesebb fizikai rekord van, Összesen 35 track-on 664 fizikai blokk van.
A kiosztás a következőképpen néz ki:

track szám	szektorok száma	rekord szám
1-17	21	1-357
18-24	19	358-471
25-30	18	478-579
31-35	17	586-664

Ilyen állományba irni, onnan olvasni a rekord számának megadásával a legkényelmesebb. Ezért az olvasáshoz ebből vissza kell számolni a track és szektor számot. A számoláshoz a következő egyszerü rutint célszerü használni:

```
1000  IF F < 358 THEN F1=0:F2=22:F3=1:
          GO TO 1040
1010  IF F > 357 AND F < 471 THEN
          F1=357:F2=20:F3=19:GO TO 1040
1020  IF F > 471 AND F < 580 THEN
          F1=471:F2=19:F3=25:GO TO 1040
1030  IF F > 579 THEN F1=579:F2=18:F3=31
1040  FT = INT ///F-F1/-1//F2-1//+F3
1050  FS = F-F1-/FT-F3/*F2+/FT-F3-1/
1100  RETURN
```

A fenti programrészben FT a track szám, FS a szektorszám, F a megadott rekord fizikai sorszáma.

A direkt elérésnél egy puffer területet használ a gép irásra/olvasásra. Igy egyszerre több állománnyal dolgozunk látszólag, mivel először a pufferbe irunk, majd onnan a lemezre. A pufferbe való irás előtt be kell állitani a puffer pointert is. Ezért például egy lemezre való irás 3 formális irással valósul meg:

- puffer pointer beállitása
- kiirás a pufferbe
- kiirás a pufferből a diskre.

A fentiek miatt mind a lemezt, mind a puffert meg kell nyitni használat előtt.
Példaképpen irjunk ki két elemet /A$ és I/ a lemez 100. rekordjába. Ezt a következő utasitássorozattal valósithatjuk meg

1. Track és szektor meghatározás:

```
F = 100 : GOSUB 1000
```

Az utasitással a fenti szubrutint felhasználva meghatározzuk a track /FT/ és a szektor /FS/ számot

2. Disk állomány megnyitása

```
OPEN 5, 8, 15
```

ahol 5 a logikai állományszám
8 a lemez egységszáma
15 a parancs és hibacsatorna

3. Puffer megnyitás

```
OPEN 4, 8, 3, "#7"
```

ahol 4 a puffer logikai száma
8 disk egységszáma
3 csatornaszám
7 pufferszám

4. Puffer pointer beállitása

```
PRINT #5 "B-P:" 3; 120
```

ahol 5 a lemez logikai száma

B-P a puffer pointer beállitását jelző parancs

3 a puffer csatornaszáma

120 a puffer pointer poziciója

5. Adatok irása a pufferbe

PRINT #4, A$; ","; I; CHR$/13/

ahol 4 a puffer logikai száma

A$,I kiirandó adatok

",",CHR$/13/ elválasztójelek

6. Adatok átvitele pufferből lemezre:

PRINT #5, "B-W:" 3; 0; FT; FS

ahol 5 az állomány /lemez/ logikai száma

B-W jelzi a pufferből-lemezre való irást

3 a puffer logikai száma

0 lemez meghajtó száma

FT track száma

FS szektor száma

A visszaolvasás ugyanilyen paraméterekkel a következőképpen történik:

7. Adatátküldés lemezről pufferbe

PRINT #5, "B-R:" 3; 0; FT; FS

ahol a paraméterek megegyeznek a 6.-nál elmondottakkal

B-R jelöli, hogy pufferbe irunk lemezről.

8. Pufferpointer beállitása

PRINT #5, "B-P:" 3; 120

ahol a paraméterek megegyeznek a 4.-nél elmondottal.

9. Adatkiolvasás pufferből a memóriába

INPUT #4, A$, I

ahol a paraméterek megegyeznek az 5.-nél elmondotakkal.

A rendszerlemezen van egy direkt állomány elérését kezelő mintaprogram. A különböző funkciók /felvitel, törlés, keresés/ tanulmányozása hozzásegit a fent vázolt funkciók megértéséhez. A mintaprogram neve RANDOM FILE.

A lemezen előforduló hibák elemzéséhez segitséget nyujt a hibacsatornán levő adatok elemzése. Ennek kiolvasását az alábbi egyszerü programmal elvégezhetjük:

```
OPEN 1, 8, 15
INPUT #1, A, B$, C, D
PRINT A, B$, C, D
```

A fenti első utasitás a 15-ös hibacsatornát nyitja meg. /Ha ez nyitva van már, természetesen nem kell megnyitni!/

A 2. utasitás innen 4 elemet olvas be, amelyek a következőt jelentik:

A hibakód sorszáma

B hibaüzenet

C track száma, ahol a hiba előfordult

D szektor szám, ahol a hiba előfordult.

Ha a fenti programrészletet beépitjük a programunkba, és minden disk müveletnél behivjuk, ugy az A-ban levő kódot elemezhetjük.

A hibakódok rövid leirását a IV. Mellékletben adjuk meg.

10,4 Sornyomtató használata

A géphez több tipusu nyomtató köthető. Mi a leggyakrabban használatos SEIKOSHA GP-100VC tipusu nyomtatóval kapcsolatos utasitásokat mondjuk el.

Mint a korábbiakban elmondtuk, a nyomtatóval kapcsolatos utasitások megegyeznek az egyéb perifériával kapcsolatos utasitásokkal /lásd 10, pont/.

A nyomtató berendezés grafikus jelek megjelenitésére is alkalmas a karakter megjelenitése mellett. Ráadásul nyomtathatunk 2 nagyságu irásjeleket /betüket, számokat stb./ is.

A nyomtatót a floppy egység /egységek/ után kell kötni legutolsó egységként. Összesen 7 különböző fokozatban állithatjuk a nyomtatás erősségét a fejen levő karra. Az egység saját maga is tartalmaz egy beépitett ellenőrzést. Ezt a gép hátoldalán lévő kapcsoló

T állásba való állitásával indithatjuk el. Megállitani a billentyü visszakapcsolásával lehet. Ugyanezen kapcsolóval állithatjuk be a nyomtató fizikai számát, amely 5 vagy 4 lehet.

10.4.1, Printer megnyitása

A használat előtt a nyomtatót meg kell nyitjunk,

Ezt a következő parancs végzi

OPEN fn, s /,sa/

ahol fn a logikai periféria név /file név/ értéke
1 és 255 közötti egész szám

s értéke 4 vagy 5 attól függően, hogy hogyan állitottuk be hátul a kapcsolót

sa második cim,

Az sa-t nem kötelező használni, automatikus értéke nulla,

A második cim értékei

0 felsőállás mód /CURSOR UP/

7 alsóállás mód /CURSOR DOWN/

A továbbiakban a printerre az fn számmal hivatkozhatunk,

10.4.2. Program kiirása nyomtatón

A programot nyomtatón is kilistáztathatjuk. Ekkor a vezérlést a printernek át kell adni. Ezt végzi a CMD utasitás.

Formája

```
CMD fn
```

ahol fn az OPEN-ben megadott azonositó

A listázás után az egységet le kell zárnunk, hogy visszaadjuk a vezérlést a gépnek.

Példa már gépben lévő program listázására:

```
OPEN 16, 4
CMD  16
LIST
```

10.4.3. Adatkiirás nyomtatón

Printeren mind programon kivül, mind programból irathatunk ki adatokat. Az első irási parancs előtt a nyomtatót meg kell nyitni.

A printerre vonatkozó kiiró utasitás:

```
PRINT #fn, l
```

ahol fn az OPEN utasitás első paramétere

l a kiirandó adatok listája.

Megjegyezzük, hogy ez az utasitás nem helyettesithető a "?" jellel /!/, mint az egyéb PRINT utasitások.

A kiirás vezérlését, módját a CHR$ utasitással vezérelhetjük.

A vezérlés a következőképpen történik:

CHR$ kód	Hatása
10	soremelés nyomtatás után
13	ugyanaz
8	grafikus módra való áttérés
14	dupla nagyságu karakter mód
15	normál karakter mód
16	kezdeti nyomtatási pozició megadása
145	felsőállás /CURSOE UP/
17	alsóállás /CURSOR DOWN/
18	forditott mező parancs
146	normál mező parancs

A továbbiakban a fontosabb kódokat vesszük végig. A grafikus módot /CHR$(8)/ az egyes 8 bitből álló oszlopok numerikus értékének DATA-ban való elhelyezésével, kinyomtatásával használhatjuk. Ezen értékeket természetesen kivülről is be lehet olvasni.

A dupla szélességü karaktereket a CHR$(14) paranccsal irathatunk ki. Ez a mód marad egészen addig, ameddig a CHR$(15) paranccsal a normál üzemmódra való áttérést el nem végeztetjük.

Például irassuk ki a FELADAT szöveget dupla nagyságu betüvel, majd a MEGOLDASA szöveget normál betünagysággal.

Ezt a következő programrész végzi el:

```
OPEN 4, 4
PRINT #4, CHR$(14) "FELADAT"
PRINT #4, CHR$(15) "MEGOLDÁSA"
```

A nyomtatás előtt megmondhatjuk, hogy a következő adat kinyomtatása melyik oszlopban kezdődjön. Ezt végzi a CHR$(16) parancs. Ennek kiadása után kétjegyű numerikus értékként meg kell adni a kezdőoszlop számát. Például irassuk ki a 25. oszloptól kezdve az ALMA szöveget.

Ezt az alábbi programrész végzi el:

```
OPEN 1,4
PRINT #1, CHR$(16) "25 ALMA"
```

Az egyes elemek kiirása után hagyandó üres helyek számát a TAB függvénnyel adhatjuk meg /lásd függvények/.

Ha az alapszint feketének, a kinyomtatott jeleket fehérnek akarjuk megadni, akkor a CHR$(18) parancsot kell használnunk.

Például a COMMODORE szöveget fekete alapon fehér betükkel a következő programrészlet irja ki

```
OPEN 5,4
PRINT #5, CHR$(18) "COMMODORE"
```

11. Grafikus megjelenitás

A gép nagyfoku grafikai képességgel rendelkezik. Kétféle lehetőség van grafikus megjelenitésre: az egyik a karakterek helyén tud megjeleniteni a billentyüzeten rögzitett grafikus jeleket, a másikkal mi magunk állithatunk elő saját alakzatokat.

Az első megoldás durvább felbontást tesz lehetővé, de egyszerüen használható. Az utóbbi megoldás esetén egész finom pontfelosztása lehetséges a képernyőnek /320 x 200 pont/.

11.1. Karakter grafika

Az egyszerübb megjelenitési formánál, mint azt fennt is mondtuk, a karakterek helyén tudunk grafikus jeleket megjeleniteni. Ekkor a képernyő fel van sorokra és oszlopokra - képzeletben egy négyzetrácsra osztva.

A képernyő mérete:

40 oszlop

25 sor

A képernyő minden egyes eleméhez - sor/oszlop karakteréhez tartozik a memóriában egy lerögzitett hely. Egy karakterhelyhez egy byte /8 bit/ tartozik, amely az aktuális helyen a képernyő tartalmát tartalmazza.

A memóriakiosztás a fentieknek megfelelően a következő:

- A képernyő "térkép" első eleme az 1024 cimü helyen van. Ez a bal felső helyet jelöli /1.sor, 1.oszlop/.
- Minden képernyő hely részére 1 byte van fenntartva. Ezek a cimek egymásutáni memóriahelyekre mutatnak, amit az alábbiakban mondunk el.
- Az elhelyezés sorfolytonosan történik egymásutáni cimekre.

A fentiekből következik, hogy az egyes elemek cimei a következők:

1. sor 1024 - 1063 cimek /összesen 40 elemnyi/
2. sor 1064 - 1103 cimek /összesen 40 elemnyi/
. . . . . .
. . . . . .
25. sor 1984 - 2023 cimek /összesen 40 elemnyi/

Minden karaktercimre összesen 256 lehetséges értéket tárolhatunk el /0-tól 255-ig/. A gép billentyüzetén levő jelek mindegyikének megfelel egy numerikus érték. Azt, hogy melyik billentyüjelhez milyen érték tartozik a III. Mellékletben foglaljuk össze.

A fenti memóriahelyekre értékeket a POKE utasitással tudunk elhelyezni. Ezért a további tárgyaláshoz szükséges az utasitás ismerete.

A POKE utasitás alakja:

POKE c, v

ahol c memóriaszám, amelyre az érték elhelyezés történik

v érték, amelyet a c cimre helyezünk el.

Mivel a fenti cimek a képernyő tartalmát adják /mármint 1024-2023/, ezért ezeknek megfelelő értéket adva bármely karaktert megjelenithetünk a képernyőn.

Például helyezzünk el a képernyő bal felső sarkába egy A betüt. Ekkor az 1024 cimre be kell tenni az A betünek megfelelő kódot, amely esetünkben az I. karakterkészletben 1 numerikus értékü.

A következő programra fogja a fenti feladatot elvégezni:

10 POKE 1024, 1

Ha A betüt akarunk iratni a bal alsó sarokba is, akkor ennek a cimére kell az 1 értéket tenni.

Ezt a következő utasitás végzi el:

20 POKE 2023, 1

A képernyő I sorához és J oszlopához tartozó számot a következő összefüggéssel határozhatjuk meg:

$$1024 + J + 40 * (I-1)$$

mivel a sorok és az oszlopok 0-tól kezdődően számozódnak.

Megjegyezzük, hogy a fenti 10, 20 utasitások hatására ha semmi sem jelenik meg a képernyőn, akkor a megjelenitett karakter szine megegyezik a képernyő szinével. Ekkor célszerü a megfelelő karakterhelyeket /nevezetesen a bal felsőt és a jobb alsót/ más szinre "befesteni". Erről a következő fejezetben beszélünk. Egy további példa egy megadott helyen "véletlen" karakterek megjelenitése a POKE utasitással:

```
10 POKE 1030, RND(X)*255:GO TO 10
```

Ha a futást le akarjuk lassitani, a két utasitás közé tegyünk egy pl. 200 lépésből álló üres ciklust.
A karakterek megjelenitéséhez a háttérszint meg kell változtatni a

```
1 POKE 55302,1
```

utasitással /lásd 12.1. pont/.

11.2. Sprite megadása

Az eddigiekben a billentyüzeten megadott karakterekkel készitettük el rajzainkat. A továbbiakban egy sokkal komplikáltabb, de több lehetőséget adó módszert foglalunk össze. Ez az un. sprite kreáció, amit az egyszerüség kedvéért a továbbiakban S betüvel fogunk jelölni.

Megjegyezzük, hogy az S megadása igen sok lehetőséget tartalmaz és sok müveletet kell elvégezni a használatához. Ezért az itteni tárgyalás csak vázolja

a lehetőségeket, a részletes leirás a gépkönyv megfelelő fejezetei /6. fejezet és N melléklet/ tárgyalják. Az S használata azért igényel több előkészületet és munkát, mert a lerajzolandó objektumot mi magunk hozzuk létre. A beosztás sokkal finomabb, igy egy sor számolást is kell végeznünk a felépitéséhez. A létrehozott objektummal egy sor egyéb müveletet is végezhetünk:

- mozgathatjuk a képernyőn
- nagyithatjuk egy, vagy mindkét irányba /vizszintes és függőleges/
- megfelelően szinezhetjük.

Ez a tipusu rajzolási lehetőség gépünk alaposabb ismeretét is igényli, ezért csak akkor ajánljuk a használatát, ha megfelelő gyakorlatra tettünk szert a programozásban, és a gépet is jobban megismertük. Egy-egy objektum ábrázolásához összesen 504 pont áll rendelkezésre. Ezt egy olyan táblázatban adhatjuk meg, amely 21 sorból és 24 oszlopból áll. Igy egy sorban összesen 24 elemről kell megadni, hogy az elem megjelenjen-e vagy sem.

A feladatot ugy oldjuk meg, hogy a 24 elemre 24 elemi tárolóhelyet, bit-et tartunk fenn, amelynek mindegyikének vagy 0 az értéke /az aktuális pont nem jelenik meg/ vagy 1 /az aktuális pont megjelenik/. A 24 bit 3 byte helyet foglal el, igy a 21 sor tárolásához összesen 63 byte értékének a kitöltése szükséges.

Az értékeket beolvashatjuk kivülről, vagy DATA utasitásban adhatjuk meg. Ha például az objektum bal felső 8 bitjére csupa sötét pontot akarunk tenni, akkor az ennek megfelelő byte-ba a következő értéket kell elhelyeznünk:

128+64+32+16+8+4+2+1 = 255

Az objektum fennt elmondott értékeinek kiszámolásához a legcélszerübb egy 21x24-es kockából álló téglalap megfelelő kockáiba felrajzolni a kivánt tárgyat, és a 63 byte-ba helyezendő értékeket ott meghatározni.

Összesen 8 független objektummal dolgozhatunk egyidőben. Az egyes objektumokra vonatkozó 63 byte-nyi táblázat cimét a memória 2040-2047 byte-jaira kell elhelyezni. A 2040-esre a 0. objektum táblázatának kezdőcimét a 2041-re az 1. objektumét ... a 20 47-re a 7. objektumét.

Az objektumokra vonatkozó fontosabb adatokat az 53248-as cimtől kezdve helyezzük el. Ezek az adatok a 8 objektumra vonatkozóan a következők lehetnek:

53248	0. objektum X koordinátája
53249	0. objektum Y koordinátája
. . .	. . . . .
53248+14	7. objektum X koordinátája

+15	7. objektum Y koordinátája
+16	legmagasabb bit X koordinátája
+21	objektum megjelenitése
+29	objektum X irányba nagyitása
+23	objektum Y irányba nagyitása
+39	0. objektum szine
+40	1. objektum szine
. . . . .	
+46	7. objektum szine

Az objektum megjelenitését adó +21. byte értékét ugy kell beállitani, hogy az

i. bit = 1 esetén az i. objektum megjelenik

i. bit = 0 esetén az i. objektum nem jelenik meg

Igy ha az 1. és az 5. objektumot akarjuk megjeleniteni, akkor a byte értékét 17-re /16+1/ kell állitanunk a POKE utasitással, azaz

POKE 2069, 17

Megjegyzendő, hogy egy byte összesen 255 értéket tárolhat, igy a képernyő oszlopai közül csak az első 255 oszlopot érhetjük el egy objektum megjelenitési céljára. Az 53264 cimü byte segitségével érhetjük el a többi oszlopot is a byte értékének megfelelő beállitásával.

Saját objektumok kreálása előtt javasoljuk a kézikönyvben megadott léggömbmozgatási példa tanulmányozását /lásd kézikönyv 6. fejezete/. Az 53248-as cimü byte-tól kezdődő memóriakiosztást a kézikönyv N melléklete tartalmazza részletesen.

12. Képernyő szinezése

A képernyő minden egyes karakterét, a képernyő alapszinét és a keretet is külön-külön szinezhetjük. Ez a látványosság mellett egy sor müszaki, tudományos, gazdasági feladat megoldásánál hasznos lehet. Ha például egy üzemben a különböző egységek bérátlagát akarjuk vizuálisan megjelentetni, akkor az átlagok kirajzolása mellé egy másik szinnel berajzolhatjuk az egész üzemi, vagy országos átlagot. Igy az egységenkénti eltérést azonnal szembeötlően lerajzoltathatjuk a géppel.

12.1. Karakterek szinezése

Mint azt a korábbiakban mondtuk, a képernyő 25 sorból és 40 oszlopból áll. Az egyes képernyő helyeken mindenegyes karakternek megadhatjuk a szinét is. Ekkor először a könyvünk elején elmondott szinkódokat elevenitsük fel:

szinkód	szin	billentyü
0	fekete	CTRL, 1
1	fehér	CTRL, 2
2	piros	CTRL, 3
3	cián	CTRL, 4

szinkód	szin	billentyü
4	bibor	CTRL, 5
5	zöld	CTRL, 6
6	kék	CTRL, 7
7	sárga	CTRL, 8
8	narancs	COMM, 1
9	barna	COMM, 2
10	világos piros	COMM, 3
11	1.szürke	COMM, 4
12	2.szürke	COMM, 5
13	világoszöld	COMM, 6
14	világoskék	COMM, 7
15	3.szürke	COMM, 8

ahol a COMM billentyü a bal alsó, Commodore jelü billentyü.

Az egyes képernyő helyek szinezéséhez egy memória hely van kijelölve, mégpedig mindenegyes helyhez egy byte tartozik. A képernyő karakterhelyeinek szintáblázata az 55296-os cimtől az 56295-ös cimig nyert elhelyezést. A táblázat a következőképpen kerül felhasználásra:

55296-55335 a képernyő első sora elemeinek szinei

55336-55375 a képernyő 2. sora elemeinek szinei

. . . . . .

. . . . . .

55256-56295 a képernyő utolsó sora elemeinek szinei

Ha például a képernyő bal felső karakterhelyének szinét akarjuk megadni, akkor az 55296-os byte értékét kell a megfelelőre beállitani. Legyen ez a szin a barna, akkor azt a következő utasitás végzi el:

```
POKE 55296, 9
```

Ezekután ezen a helyen megjelenő bármely karakter szine barna lesz.

Példaképpen fessük a képernyő első 15 sorát különböző szinüre. A fennt megadott szinek következzenek egymás után soronként.

A feladatot a következő programrész fogja elvégezni:

```
10 FOR J = 0 TO 14
20 FOR K = 0 TO 39
30 POKE 55296 + J * 40 + K, J + 1
40 NEXT K
50 NEXT J
```

Ha a szinezést be is akarjuk mutatni, akkor valamilyen karaktert meg is kell jelentetni a megfelelő helyeken. Ezt végzi a következő programrész

```
 60  PRINT "SHIFT, CLR"
 70  FOR J = 0 TO 14
 80  FOR K = 1 TO 39
 90  POKE 1024 + J * 40 + K, 224
100  NEXT K
110  NEXT J
```

12.2. Egyéb szinezési lehetőségek

A képernyő karakterhelyeinek szinezése mellett a képernyő alapszinét és a keret szinét is megadhatjuk. A képernyő alapszinét az 53281-es byte tartalmazza. Igy az alapszint a

```
POKE 53281, I
```

utasitással lehet beállitani, ahol I 0 és 15 közötti egész érték lehet.
A keret szinének megadására is lehetőség van. Ezt az 53280-as byte tartalmazza.
A keretszin a

```
POKE 53280, I
```

utasitással állithatjuk be, ahol I a fentiekben elmondott.
A képernyő összes alap és keretszinének kombinációját állithatjuk össze a következő programrészlettel:

```
10 FOR I = 0 TO 15
20 POKE 53281, I
30 FOR J = 0 TO 15
40 POKE 53280, J
50 FOR K = 1 TO 400 : NEXT
60 NEXT J
70 NEXT I
```

13. Zenei lehetőségek

A számitógép igen hatékony zenei szintetizátorként is használható. Egyszerre 3 hang szólaltatható meg, mindegyik külön szólamban és összesen 9 oktávnyi terjedelemben. A fentiek mellett változtatható a

- hullámforma /4 féle/
- hangerősség
- hang "hosszusága"
- szimulálható a zeneeszköz /összesen 6 féle/
- egyéb hanggal kapcsolatos jellemzők.

A zene megszóleltatható BASIC nyelven belül és természetesen gépi nyelvü programmal is. A gyár elkészitett egy programot a különböző lehetőségek tanulmányozására /a program neve: SIDMON/, amellyel 19 féle változó beállitására van lehetőségünk.

A fontosabbakat az alábbiakban összegezzük:

Hang jellemző	Tartomány /min, max/
Frekvencia	0,57344
Pulzálás	0,4032
Felfutás nulláról	0,15
Lecsengés nullára	0,15

Átlag hangerősség	0,15
Átlagszintre leereszkedés	0,15
Hang gátlás /kibocsájtás/	0,1
Két hullámforma szinkronizációja	0,1
Gyürü moduláció	0,1
Hullámforma	1,4
Szürő /felharmónikusok/	0,1
Rezonancia	0,15
Hangerősség	1,15
Frekvencia vágás	0,232
Hang	1,3

A fenti értékekre való állitás egyesével folyamatosan történhet a két érték között. A hangot a 6581-es csip- a hang és zenei szintetizátor hozza létre.

Ez a 6500-as mikroprocesszor családhoz köthető önálló egység /un. SID csip/.
A főbb jellemzői a következők:

- 3 hang oszcillátor /0-4 kHz között mindegyik/
- 4 hullámforma oszcillátorként /háromszög, fürészfog, pulzáló, zaj/

- 3 amplitudó modulátor /48 dB terjedelem/
- 3 hangvariációs generátor

felfutás: 2 ms - 8 np

lecsengés: 6 ms - 24 mp

átlagos szint: 0 - max.

átlagszintre ereszkedés: 6 ms - 24 mp

- oszcillátor szinkronizáció
- gyürü moduláció
- programozható szürő

vágási tartomány 30 Hz - 12 kHz

- hangerősség szabályzó
- két analóg /digitál potenciométer csatlakoztatási lehetőség
- véletlen szám - modulációs generátor
- külső hang bemenet.

A különböző lehetőségek vezérléséhez 29 byte-os regiszter áll rendelkezésre. Ezek egy része ROM /4 byte/, másik része RAM.

Zenei paraméterek beállitása

A zenei paramétereket az előzőekben összefoglaltuk. Ezek mindegyikének megfelel egy memória-

hely, amelynek értékeit a POKE utasitással állithatjuk be.

A SID csip regisztereinek megfelelő memóriahelyek az 54272-54296 cimeken találhatók.

A fontosabb memóriahelyek a következők:

hangmagasság /frekvencia: 0 - 65000/ :

alsó byte: 54272

felső byte: 54272 + 1

hangerősség /0 és 15 közötti érték/:

54272 + 24 = 54296

hangszin /hullámforma/:

54272 + 4 = 54276

értékei: 17 háromszög

33 fürészfog

65 pulzáló

129 zaj

felfutás nulláról /lecsengés nullára

54272 + 5 : 54277

értékei: 15x16 - 0x16 /felfutás/

/felső négy bit felfutás esetén

15 - 0 /lecsengés/

/alsó négy bit lecsengés esetén

átlagerősség/átlagszintre leereszkedés

54272 + 6 = 54278

értékei: 15 *16 - 0*16 /átlagerősség/

/felső négy bit átlagerősség esetén

15 - 0 /leereszkedés/

/alsó négy bit átlagszintre leereszkedés esetén./

A fenti értékeket az 1.hang esetére adtuk meg.

Az alábbi táblázatba összefoglaljuk a fontosabb jellemzők megadásának memóriahelyeit mindhárom hang esetén /S=54272/

	1.hang	2.hang	3.hang
hangmagasság			
alsó byte	S	S + 7	S + 14
felső byte	S + 1	S + 8	S + 15
hullámforma	S + 4	S +11	S + 18
felfutás/lecsengés	S + 5	S +12	S + 19
átl.e,/a.leer.	S + 6	S +13	S + 20

A zenei lehetőségek a fentieknél lényegesen többet tartalmaznak, részletes leirásukat a gyári kézikönyv tartalmazza.

A hangmagasság beállitására 3 oktáv esetén adjuk meg a felső és alsó byte értékeit a VI.Mellékletben.

Példaképpen bemutatunk egy "computer"-zenét szolgáltató programot. A látszólag ösze-vissza hangokat véletlenszám generátorral állitottuk elő. A computer zene programja a következő:

```
10 B = 54272
20 FOR T = B TO B + 24
30 POKE T, 0
40 NEXT T
50 POKE B + 4, 17
60 POKE B + 5, 255
70 POKE B + 6, 255
80 FORTL I = 1 TO 140
90 POKE B +1, RND(1)*215
100 POKE B,RND (2)*235
110 POKE B + 24, 15
120 FOR T = 1 TO 50 : NEXT
130 NEXT
140 POKE B + 24,0
```

A fenti prcgram utasitásait vegyük sorra.

A 10 utasitásban beállitjuk a bázis cimet. Erre tulajdonképpen nincs szükség, de egyszerüsiti a programirást.

A 20 - 40 utasitásban kinullázzuk a zenei paramétereket tartalmazó memóriahelyeket. Erre abban az

esetben nincs szükség, ha korábban nem használtuk ezen memóriarészt /mivel lekapcsoláskor az összmemória tartalma nulla/.

Az 50 utasitás a hangszint /hullámformát/ állitja be az egyes hang esetén háromszögre.

A 60 utasitás a felfutás/lecsengés értékét állitja be 255-re. A 70 utasitás az átlagerősség/átlagszint-értékét állitja be.

A 80-130 utasitássorozat összesen 280 véletlen számot generál, amelyek a hangmagasság alsó és felső byte-jának beállitására szolgálnak.

Az alsó byte értékének beállitását a 90, a felsőét a 100 utasitás végzi.

A 110 utasitással a hangerőt állitjuk be legerősebbre. Ha a hangerőt is változtatni akarjuk hangonként akkor az alábbi utasitást kell beirni a 110 helyébe

```
110 POKE B + 24,RND(3)*15
```

A 120 utasitás egy hang hangzásának hosszuságát állitja be /tulajdonképpen egy 50 elemnyi üres ciklus/

A 140 utasitás a hangerőt kapcsolja le nullára.

A másik, szintén szférikus zenét előállitó programot a különböző variálási lehetőségek bemutatására közöljük,

```
10 B = 54272
20 FOR T = B TO B + 24
```

```
30 POKE T,0
40 NEXT T
50 POKE B+5,255
60 POKE B+6,255
70 FOR E=1 TO 2
 80 POKE B+24,15:K=K+1
 90 POKE B+4,65:J=14
100 POKE B+3,15
110 POKE B+2,255
120 FOR T=1 TO 242
130 F = INT(RND(1)*10)
140 D=INT(RND(1)*10)
150 IF D < 2  THEN J = INT(J/2)
160 IF J-F < = 0  OR J-F = 255 THEN
    J=50: GO TO 200
170 POKE B+1,J-F
180 POKE B+3,INT(T/16)
190 J=J+1
200 NEXT T,E
210 POKE B+24,0
```

A fenti program hasonló az előzőhöz, ezért a részletes elemzésétől eltekintünk.

Egy kedves zenének a programját is elkészitettük.

Ez az "Egy kis malac" c. gyermekdal első két sora.

A program a következő:

```
   10 B=54272
   20 FOR T=B TO B+24
   30 POKE T,0
   40 NEXT T
   50 POKE B+5,255
   60 POKE B+6,20
   70 POKE B+24,15
   80 POKE B+4,17
   90 FOR I=1 TO 33
  100 POKE B+4,17
  110 READ J,K
  120 POKE B,K:POKE B+1,J
  130 FOR L=1 TO 150:NEXT
  140 POKE B+4,16
  150 NEXT I
  160 DATA  0,0,0,0,0,0
  170 DATA 21,31,0,0,25,30,0,0,18,209,0,0
  180 DATA 21,31,0,0,17,195,0,0,17,195,0,0
  190 DATA 17,195,0,0,0,0,21,31,0,0,25,30,0,0
  200 DATA 18,209,0,0,21,31,0,0,17,195,0,0
  210 DATA 17,195,0,0,17,195,0,0,0,0
```

A fenti programot elinditva a hangok "nem voltak a helyükön", Ezért megpróbáltuk "hangolni".

Ehhez a következő változtatásokat végeztük a programon:

a/ 110 POKE B,K-1:POKE B+1,J-3

b/ DATA utasitásokban a hangokat elválasztó 0,0 értékek helyett 3,1 párt ütöttünk be.

Igy például a 160 utasitás a következőképpen néz ki:

DATA 21,31,3,1,25,30,3,1,18,20:,3,1

Javaşoljuk mindkét változat kipróbálását!

I.sz.Melléklet

C 64 Hibaüzenetek

A fejezet a hibaüzenetek teljes listáját adja meg.

BAD DATA ... Egy nyitott adatállományból karakter típusu adatot olvasott be a gép, de numerikus adatot várt.

BAD SUBSCRIPT ... A gép egy olyan mátrixelemre hivatkozott, amely a DIM által megadott elemhatáron kivül fekszik.

BREAK ... Programvégrehajtást a STOP billentyüvel megszakitottuk.

CAN'T CONTINUE ... A CONT parancs nem hajtódik végre. Ennek egyik lehetséges oka: eddig még nem adtuk ki a RUN parancsot, hiba van a programban, stb.

DEVICE NOT PRESENT ... A gép által kivánt egység nincs a géphez kötve /OPEN, CLOSE CMD, PRINT ≠ INPUT ≠, GET ≠ utasitások esetén adódik ez az üzenet/

DIVISION BY ZERO ... Zéróval való osztás történt

EXTRA IGNORED ... Tul sok adat lett begépelve egy INPUT utasitás esetén, a feleslegeseket a program nem vette figyelembe. A futás tovább folytatódik.

FILE NOT FOUND ... Ha az adatállományt szalagon keresték, az állomány végét /END OF TAPE/ nem találta. Disk esetén a megadott nevü állomány /program/ nincs a gépben levő lemezen.

FILE NOT OPEN ... Az adatállományt először meg kell nyitni, mielőtt a CLOSE, CMD, PRINT ≠, INPUT≠, GET≠ utasitások valamelyikét használnánk.

FILE OPEN ... Egy olyan adatállományt kisérelt meg-nyitni, amelyik már korábban nyitva volt.

FORMULA TOO COMPLEX ... A karakter tipusu kifejezés tul összetett, legalább két részre szét kell szedni. Ezt az üzenetet kapjuk akkor is, ha egy kifejezésben tul sok a záró-jel.

ILLEGAL DIRECT ... Az INPUT utasitást csak programon belül lehet használni, direkt módban nem.

ILLEGAL QUANTITY ... Az utasitásban a függvény független változójaként használt érték nem megfelelő.

LOAD ... Probléma van a szalagon lévő programmal.

NEXT WITHOUT FOR ... Vagy nem megfelelő helyen szerepel a NEXT utasitás /pl. FOR előtt/, vagy a NEXT változója nem egyezik meg a ciklusváltozóval.

NOT INPUT FILE ... Egy olyan adatállományt akartunk olvasni INPUT vagy GET utasitással, amely csak outputként használható.

NOT OUTPUT FILE ... Egy olyan adatállományba akartunk irni, amely csak inputként használható.

OUT OF DATA ... Egy READ utasitást akart végrehajtani, de a DATA utasitásban már nem volt adat.

OUT OF MEMORY ... Nincs több hely a memóriában valamely adat vagy program számára. Ez akkor is bekövetkezhet, ha tul sok FOR ciklus van egymásba skatulyázva, vagy tul sok GOSUB utasitás aktiv egyszerre.

OVERFLOW ... A számitás eredménye nagyobb, mint a lehetséges legnagyobb szám, amit tárolhatunk a gépben.

REDIM'D ARRAY ... Egy tömb csak egyszer lehet dimenzionálva. Ha egy tömbbel a rá vonatkozó DIM utasitás előtt találkozik, akkor automatikusan deklarálja a gép, mint 10 elemü vektort vagy magasabb dimenziós tömböt. Ugyanezen tömbre később kiadott DIM utasitás szintén ezt a hibát hozza.

REDO FROM START ... Egy karakter tipusu adatot gépeltünk be az INPUT utasitás hatására, de a gép numerikus tipusu adatot várt. A futás az üzenet hatására nem áll le, hanem tovább inditható megfelelő tipusu adat begépelésével.

RETURN WITHOUT GOSUB ... A futás során egy RETURN utasitást ért el a gép, de nem GOSUB utasitással jutottunk el a programrészre. Ez rendszerint akkor fordul elő, ha GO TO utasitással ugrottunk ide, vagy a szubrutin nem volt "levédve" és a program végrehajtás során nem kerülte ki a szubrutint.

STRING TOO LONG ... A karakter tipusu konstans hosszabb, mint 255 karakter.

SYNTAX ... A hiba nem felismerhető a gép által.
Ilyen üzenetet kapunk például, ha több nyitó zárójel van vagy elmarad, nem megfelelő a kulcsszó stb.

TYPE MISMATCH ... Az üzenet akkor fordul elő, ha numerikusat használunk karakter tipus helyén, vagy forditva. /karakter tipusut numerikus helyett./

UNDEF'D STATEMENT .. Egy olyan utasitáscimke szerepel a GOTO, GOSUB vagy RUN után, amely nem fordul elő a programban.

VERIFY ... A szalagon vagy lemezen levő program nem egyezik meg a memóriában levő programmal.

II.sz.Melléklet

STATUS függvény értékei

ST bit pozició	ST numerikus értéke	Kazetta olvasás	Soroshan olvasás/irás	Szalag /VERIFY/ betöltés	RS 232 adat-csatorna
0	1	-	idön tuli irás	-	paritáshiba
1	2	-	idön tuli olvasás	-	eszközhiba
2	4	rövid blokk	-	rövid blokk	fogadó puffer betelt
3	8	hosszu blokk	-	hosszu blokk	fogadó puffer üres
4	16	beazonositha-tatlan olvasási hiba	-	bármilyen adatösszeférhetetlenség	CTS jel bit nincs
5	32	ellenőrző összeghiba /chechkum/	-	ellenőrző összeghiba	nem használt
6	64	állomány vége	EOI	-	DSR jel bit nincs
7	128	szalag vége	egység nincs jelen	szalag vége	megszakitás

III. Melléklet

Karakter kódok
POKE utasitáshoz

I. készlet	II. készlet	POKE	I. készlet	II. készlet	POKE
@		0	Q	q	17
A	a	1	R	r	18
B	b	2			
C	c	3	S	s	19
D	d	4	T	t	20
E	e	5	U	u	21
F	f	6	V	v	22
G	g	7			
H	h	8	W	w	23
I	i	9	X	x	24
J	j	10	Y	y	25
K	k	11	Z	z	26
L	l	12	[		27
M	m	13	£		28
N	n	14	]		29
O	o	15	↑		30
P	p	16	←		31
sp		32	1		49
!		33	2		50
			3		51
"		34	4		52
#		35			
			5		53
$		36	6		54
%		37	7		55
&		38	8		56
'		39	9		57
			:		58
(		40			
			;		59
)		41			

I.készlet	II. készlet	POKE	I.készlet	II. készlet	POKE
*		42	<		60
			=		61
+		43			
`		44	>		62
-		45	?		63
.		46	⊟		64
/		47	Sziv	A	65
0		48	⊞	B	66
JG	C	67	JG	T	84
JG	D	68	JG	U	85
JG	E	69	JG	V	86
JG	F	70	JG	W	87
JG	G	71	JG	X	88
JG	H	72	JG	Y	89
JG	I	73	JG	Z	90
JG	J	74			
JG	K	75			
JG	L	76			
JG	M	77			
JG	N	78			
JG	O	79			
JG	P	80			
JG	Q	81			
JG	R	82			
JG	S	83			

Megjegyzések:

1./ A fenti táblázatban csak az egyszerüen azonositható karaktereket soroltuk fel.
A teljes karakterkészletből a 0-127-hez tartozó kódokat a gyári kézikönyv E Melléklete tartalmazza.

2./ Az inverz kódokat ez a könyv nem tartalmazza, de könnyen előállithatók az eredeti kódok 128-al való megnövelésével és kiiratásával.

3./ A JG rövidités a megfelelő billentyü jobb alsó oldalán levő grafikus jelre utal.

IV.sz.Melléklet

Disk hibaüzenetek

Hibakód	Üzenet
0	Nincs hiba
1	SCRATCH parancs hibátlanul lement
2-19	- nem használt üzenetek
20	READ ERROR ... Blokk fejet nem talált a lemezen
21	READ ERROR ... Nem talált szinkron jelet az aktuális track-en
22	READ ERROR... Nincs megfelelően kiirt adatblokk olvasás vagy "verify" esetén
23	READ ERROR .... Ellenőrző összeg hiba. Egy vagy több adatot beolvasott, de az ellenőrző összeg nem megfelelő
24	READ ERROR ... Nem megengedett byte-t talált az olvasás közben
25	WRITE ERROR ... A kiirt és a memóriában levő adat nem egyezik meg /pl. verify esetén/.
26	WRITE PROTECT ON ... Irni akar, de az irás gomb nincs benyomva./pl. disketten az irás négyzet le van ragasztva/.
27	READ ERROR ... Az adatblokk fejében hibát talált a vezérlő.
28	WRITE ERROR ... Hosszu adatblokk.
29	DISK ID MISMATCH ... Olyan disketten dolgozna, amely nem lett inicializálva, vagy a diskett cimke rossz.

30 SYNTAX ERROR ... A DOS nem tudja azonositani a parancs csatornán küldött parancsot /pl. illegális file név stb./

31. SYNTAX ERROR ... A DOS a parancsot nem tudja felismerni, ha nem az első pozicióban kezdődik.

32 SYNTAX ERROR ... A küldött parancs hosszabb 58 karakternél.

33 SYNTAX ERROR ...Nem megengedett file nevet használunk egy OPEN, SAVE parancsban.

34 SYNTAX ERROR ... A file név kimaradt vagy nem azonositható a parancsban /pl. "." hiányzik/.

39 SYNTAX ERROR ... Nem megengedett parancsot küldtünk a 15-ös parancs csatornán.

50 RECORD NOT PRESENT ... Az utolsó rekord után olvassunk INPUT vagy GET utasitással.

51 OVERFLOW IN RECORD ... A PRINT utasitásban tul hosszu adatsort akarunk kiirni. Az információ csonkitódik.

52 FILE TOO LARGE ... A file-ben több rekord nem fér el.

60 WRITE FILE OPEN ... Egy irásra megnyitott állományt nem zártunk le az olvasás elött.

61 FILE NOT OPEN ... Az adatállománnyal megnyitás nélkül kisérelünk meg dolgozni.

62 FILE NOT FOUND ... Az aktuális lemezegységen a keresett adatállomány nincs rajta.

63 FILE EXISTS ... A most létrehozandó file azonos névvel már létezik.

64 FILE TYPE MISMATCH ... Az adatállomány tipusa nem megfelelő.

65 NO BLOCK ... A most allokálandó állományt már egyszer allokáltuk.

66 ILLEGAL TRACK AND SECTOR ..Egy olyan track-kel és szektorral akarunk dolgozni, amely nem létezik a megadott formánál.

67 ILLEGAL SYSTEM T OR S ... Illegális track és szektor.

70 NO CHANNEL ... A kiválasztott csatorna nem áll rendelkezésre, illetve az összes csatorna foglalt /egyszerre max. 5 file lehet megnyitva ! /

71 DIR ERROR ... A blokk foglaltási térképpel /BAM/ problémák vannak.

72 DISKFULL ... Vagy a blokk a disketten foglalt, vagy a direktori betelt.

73 CBM DOS V26V170 ... A különböző DOS verziók nem keverhetők .

A 1541-es egységen 2.6 DOS és 1.0 verzió /CBM 2040/3040 / olvasás kompatibilisek, de nem irás kompatibilisek.

A DOS 1.0 alatt lehet olvasni 1541-en, de nem lehet irni.

74 DRIVE NOT READY ... A 1541-es egységen dolgozni akarunk, de nincs disk benne.

V.sz.Melléklet

Fontosabb memóriahelyek

Decimális Cim	Leirás
0	6510-es adatirány regiszter
1	I/10 regiszter /Konzol ROM, kazetta motor kontroll/
2	Nem használt
3-4	Ugrás vektor: konvertálás lebegőpontosról egészre
5-6	Ugrás vektor: konvertálás egészről lebegő-pontra
7	Keresés karakter
8	Mutató: string végidőjele
9	Az utolsó TAB képernyő oszlopa
10	Mutató: 0 = Load, 1 = Verify
11	I/10 puffer printer
12	Mutató: hibás tömb dimenzió
13	Adattipus /string v. numerikus/
14	Adattipus /egész, valós/
17	Mutató: INPUT, GET vagy READ
43-44	Mutató: BASIC kezdete
45-46	Mutató: BASIC változók kezdete
47-48	Mutató: BASIC tömbök kezdete
49-50	Mutató: BASIC tömbök vége
51-52	Mutató: string tároló vége
55-56	Mutató: BASIC legmagasabb cim
57-58	Aktuális BASIC sorszám
59-60	Elöző BASIC sor száma
61-62	Mutató: A CONT helye a BASIC-ben
63-64	Aktuális DATA sorszáma
71-72	Aktuális BASIC változó
73-74	Mutató: Ciklusváltozó

97	Lebegőpontos akkumulátor: kitevő
98-101	Lebegőpontos akkumulátor: karakterisztika
113-114	Mutató: kazetta puffer
145	Mutató: STOP kulcs /RVS kulcs
147	Mutató: 0 = Load, 1 = Verify
152	Nyitott file-k száma /file táblázat index
157	Mód: Direkt M., Program M.
165-171	RS232 I/O byte-ok
183	Aktuális file név hossza
184	Aktuális logikai file szám
186	Aktuális egységszám
193-194	I/O kezdőcim
201-202	Kurzor X-Y pozíciója
208	Mutató: INPUT vagy GET a billentyüzetről
213	Aktuális képernyősor hossza
243-244	Mutató: Aktuális képernyő szin RAM hely
247-250	RS-232 I/O puffer printer
256-511	Mikroprocesszor rendszer verem területe
512-610	Rendszer INPUT puffere
601-630	KERNAL táblázat: file számok és egységszámok
643	Aktuális karakter szin kód
648	Képernyő memória kezdő cime
659-670	RS-232 mutatók
679-767	Nem használt
780	Processzor A regisztere
781	" X "
782	" Y "
783	" SP "
788-789	Hardware IRQ interrapt vektor

794-819	KERNAL rutinok vektorai
820-827	Nem használatos
829-1019	Szalag I/IO puffer
1020-1023	Nem használatos
1024-2023 /2047/	Képernyő memória terület
2040-2047	Sprite adat - mutató
2048-40959	Normál BASIC terület
32768-40959	VSP Carridge ROM /8192 byte/
40960-49151	BASIC ROM - 8192 byte /vagy 8 K RAM/
49152-53247	4096 byte RAM
53248-57343	I/O egységek és szines RAM vagy karakter generátor ROM vagy RAM /4096 byte/
57344-65535	KERNAL ROM /8194 byte/

I/O utasitások fontosabb helyei

Decimális cim	Leirás
53248-54271	MOS 6566 VIDEO interface kontroller
53248-53264	Sprite 0-7 XY pozició
53265	VIC kontroll regiszter
53266	READ/WRITE raszter
53267-53268	Fényceruza X,Y
53271	Sprite kétszeres kiterjesztés Y irányába
53280	Keretszin
53281-53284	Háttérszinek
53287-53294	Sprite 0-7 szin
54272-55295	MOS 6581 zenei interface
55296-56312	Color RAM
56320-56575	MOS 6586 CIA /Complex Interface Register/ 1

56324-56331	Idő regiszterek
56576-56831	MOS 65 26 CIA 2
56580-56587	Idő regiszterek
56832-57087	Foglalt a leendő I/O kiterjesztéshez
57088-57343	Foglalt a leendő I/O kiterjesztéshez

VI.Melléklet

Hangmagasság beállitása

A beállitásra az 54272 /alsó byte/ és 54273 /felső byte/ szolgál. A különböző hangok értékei a középső 3 oktávra a következők:

Oktáv	Hangmagasság		
	decimális	felső byte	alsó byte
C-3	2145	8	97
C#3	2273	8	225
D-3	2408	9	104
D#3	2551	9	247
E-3	2703	10	143
F-3	2864	11	48
F#-3	3034	11	218
G-3	3215	12	143
G#-3	3406	13	78
A-3	3608	14	24
A#-3	3823	14	239
B-3	4050	15	210
C-4	4291	16	195
C#-4	4547	17	195
D-4	4817	18	209
D#-4	5103	19	239
E-4	5407	21	331
F-4	5728	22	96

Oktáv	Hangmagasság		
	decimális	felső byte	alsó byte
F#-4	6069	23	181
G-4	6430=	25	30
G#-4	6812	26	156
A-4	7217	28	49
A#-4	7647	29	223
B-4	8101	31	165
C-5	8583	33	135
C#-5	9094	35	134
D-5	9634	37	162
D#-5	10207	39	223
E-5	10814	42	62
F-5	11457	44	193
F#-5	12139	47	107
G-5	12860	50	60
G#-5	13625	53	57
A-5	14435	56	99
A#-5	15295	59	190
B-5	16203	63	75

VII.Melléklet

CP/M operációs rendszer használata

A COMMODORE operációs rendszer /OS/ sok lehetőséget tartalmaz. Egyrészt rendkivül sok periféria köthető közvetlenül a géphez, és ezen berendezések használata nem igényel extra memóriát, mivel a számitógép perifériái intelligensek. Másik előnye az OS rendszernek, hogy felfelé kompatibilisek, ami azt jelenti, hogy kisebb gépeken kifejlesztett rendszerek nagyobb, vagy a későbbiekben kijövő COMMODORE gépeken is futtathatók. Egyetlen hátránya az OS-nek, hogy más gépen megirt programok nem futtathatók a berendezésen. Ez a legfőbb ok, ami miatt elkészitette a cég a CP/M kártyát, amellyel más, CP/M operációs rendszerrel rendelkező gépeken megirt programot is elfogad a számitógépünk.

A Z80 mikroprocesszorral felszerelt CP/M kártya /kartridge/ közvetlenül bedugható a gép hátoldalán elhelyezett csatlakozóba. A kártyához egy, a CP/M operációs rendszert tartalmazó floppy lemez is szükséges.

A CP/M futtatása a következőképpen történik:

1. Töltsük be a CP/M rendszert a

LOAD "CP/M",8 paranccsal.

2. Futtassuk a rendszert:

gépeljük be a RUN és RETURN szót.

Ettől kezdve a gép 64K RAM memóriája rendelkezésre áll, de a Z-80 kártya 48K RAM memóriát is használhatjuk. A lényeg, hogy a két memória közül egy programon belül csak az egyik RAM érhető el a fenti lehetőségek közül.

A memória cimzése:

a Z80 RAM cimek 4096-al kisebb cimen

vannak, mint a COMMODORE cimek.

Pontosabban a következő cimek felelnek meg egymásnak:

Z - 80	6510
0000 - 61439	4096 - 65535
61440 - 65535	0000 - 4095

A CP/M kártya használata a Z 80 Reference Guide cimü könyvben van részletesen leirva.

VIII.Melléklet

BASIC kulcsszavak, parancsok és rövidítéseik

Az alábbiakban felsoroljuk a COMMODORE 64 BASIC alapszavakat, a parancsokat és ezek rövidítését. Megjegyezzük, hogy a rövidítést használva a program nem foglal el kevesebb helyet, mivel minden egyes kulcsszó összesen egy byte-ban kerül tárolásra.

Parancs	Rövidités	Függvény tipus
ABS	A S B	numerikus
AND	A S N	
ASC	A S S	numerikus
ATN	A S T	numerikus
CHR$	C S H	karakter
CLOSE	CL S O	
CLR	C S L	
CMD	C S M	
CONT	C S O	
COS	-	numerikus
DATA	D S A	
DEF	D S E	
DIM	D S I	
NOT	N S O	
ON		
OPEN	O S P	
OR		

Parancs	Rövidités	Függvény tipus
PEEK	P S E	numerikus
POKE	P S O	
POS	-	numerikus
PRINT	?	
PRINT#	P S R	
READ	R S E	
REM	-	
RESTORE	RE S S	
RETURN	RE S T	
RIGHT$	R S I	karakter
RND	R S N	numerikus
RUN	R S U	
SAVE	S S A	
SGN	S S G	numerikus
SIN	S S I	numerikus
S P C	S S P	
S Q R	S S Q	numerikus
STATUS	ST	numerikus
END	E S N	
EXP	E S X	numerikus
FN	-	
FOR	F S O	
FRE	F S R	numerikus
GET	G S E	

Parancs	Rövidités	Függvény tipus
GET≠	-	
GOSUB	GO S S	
GOTO	G S O	
IF	-	
INPUT	I S N	
INT	-	numerikus
LEFT$	LE S F	karakter
LEN	-	numerikus
LET	L S E	
LIST	L S I	
LOAD	L S O	
LOG	-	numerikus
MID	M S I	karakter
NEW	-	
NEXT	N S E	
STEP	ST S E	
STOP	S S T	
STR$	ST S R	karakter
SYS	S S Y	
TABL	T S A	
TAN	-	numerikus
THEN	T S H	
TIME	TI	numerikus

Parancs	Rövidités	Függvény tipus
TIME$	T I $	karakter
TO	-	
USR	U S S	numerikus
VAL	V S A	numerikus
VERIFY	V S E	
WAIT	W S A	

Megjegyzés:

A fenti felsorolásban a rövidítéseknél a középső S betü a SHIFT billentyü lenyomását jelöli.

TARTALOM

oldal

1, Gép felépitése, főbb jellemzői 1

1.1. Memória 1

1.2. Képernyő 2

1.3, Hangeffektusok 4

1.4, Kapcsolható perifériák 6

1,4.1. Kazettás egység..... 6

1.4.2. Nyomtató berendezés.....7

1.4.3. Mágneslemez egység.....9

1,4.4, Egyéb perifériális egységek.....10

2. Gép billentyüzete, csatlakozói.....12

2.1. Bekapcsolás.....12

2.2. Gép csatlakozói.....13

2.3. Billentyüzet.....18

3. BASIC programok irása.....22

4. BASIC alapelemei.....25

4.1, BASIC alapjelek.....25

4.2. Konstansok.....25

4.2.1. Egész és valós konstansok.....26

4.2.2. Karakter konstans.....29

4.3. Változók.....29

4.3.1. Változók tipusa.....30

4.3.2. Tömbök megadása.....31

oldal

4.4. Aritmetikai kifejezés és értékadó utasitás.... 33

4.4.1. Aritmetikai kifejezés felépitési szabályai..... 33

4.4.2. Aritmetikai kifejezés tipusa..... 36

4.4.3. Értékadó utasitás..... 37

4.5. Reláció és logikai kifejezés..... 40

4.5.1. Reláció..... 40

4.5.2. Logikai kifejezés..... 41

4.6. Input-output utasitások..... 42

4.6.1. Beolvasó utasitások..... 43

4.6.1.1. INPUT utasitás..... 43

4.6.1.2. Egy karakter vagy karakterso-

rozat gépbe vitele..... 46

4.6.1.3. Beolvasás a READ utasitással. 47

4.6.2. Kiiró utasitások..... 51

4.6.2.1. PRINT utasitás..... 51

4.6.2.2. Kiirás helyének vezérlése ka-

rakterekkel..... 53

4.6.2.3. Forditott szin és szines vezér-

lés..... 54

4.6.2.4. Kiirás vezérlése SPC, TAB uta-

sitásokkal..... 56

4.7. Vezérlő utasitások..... 58

4.7.1. IF feltételes utasitás..... 58

4.7 2. GO TO ugró utasitás..... 59

oldal

4.7.3. FOR ciklusutasitás..... 62

4.7.4. STOP és END utasitások..... 69

5. Programok kezelése..... 69

5.1. Futtatás, megállitás, ellenőrzés..... 69

5.2. Módositás, listázás, javitás, törlés..... 70

6. Program tárolása és betöltése háttértárolóból..... 74

6.1. Program elhelyezése háttértárolón..... 74

6.2. Program kivitel ellenőrzése..... 77

6.3. Program betöltése háttértárolóból..... 78

6.4. Szabad memóriahely ellenőrzése..... 81

6.5. Irás/olvasás ellenőrzése..... 81

7. Karaktersorozatokkal kapcsolatos müveletek..... 83

7.1. Értékadás..... 83

7.2. Karakter tipusu változók összeadása..... 84

7.3. Karakterekre vonatkozó függvények..... 85

8. Matematikai és egyéb függvények..... 91

8.1. Matematikai függvények..... 91

8.2. Véletlen szám generátor 93

8.3. Időmérő függvények..... 94

8.4. Saját függvény megadása..... 96

9. Szubrutinok készitése,..... 98

10. Háttértárak használata adattárolásra..... 101

10.1. Kazettás magnetofon használata..... 102

10.1.1. Adatállomány létrehozása..... 102

10.1.2. Adatkivitel kazettára..... 105

10.1.3. Adatállomány lezárása..... 108

10.1.4. Beolvasás kazettáról 109

Oldal

10.2. Egyéb perifériák kezelése..... 110
10.2.1. Perifériák megnyitása..... 110
10.2.2. Irás/olvasás..... 112
10.2.3. Egységek lezárása..... 114
10.3. Mágneslemez használata..... 114
10.3.1. Rendszerlemez programjai..... 115
10.3.2. Soros állománykezelés..... 118
10.3.3. Direkt állomány..... 121
10.4. Sornyomtató használata..... 126
10.4.1. Printer megnyitása..... 127
10.4.2. Program kiirása nyomtatón..... 128
10.4.3. Adatkiirás nyomtatón..... 128
11. Grafikus megjelenités..... 131
11.1. Karakter grafika..... 131
11.2. Sprite megadása..... 134
12. Képernyő szinezése..... 138
12.1. Karakterek szinezése..... 138
12.2. Egyéb szinezési lehetőségek..... 141
13. Zenei lehetőségek..... 142

M E L L É K L E T E K

I. C64 hibaüzenetek 152

II. STATUS függvény értékei 156

III. Karakter kódok POKE utasitáshoz 157

IV. Disk hibaüzenetek 160

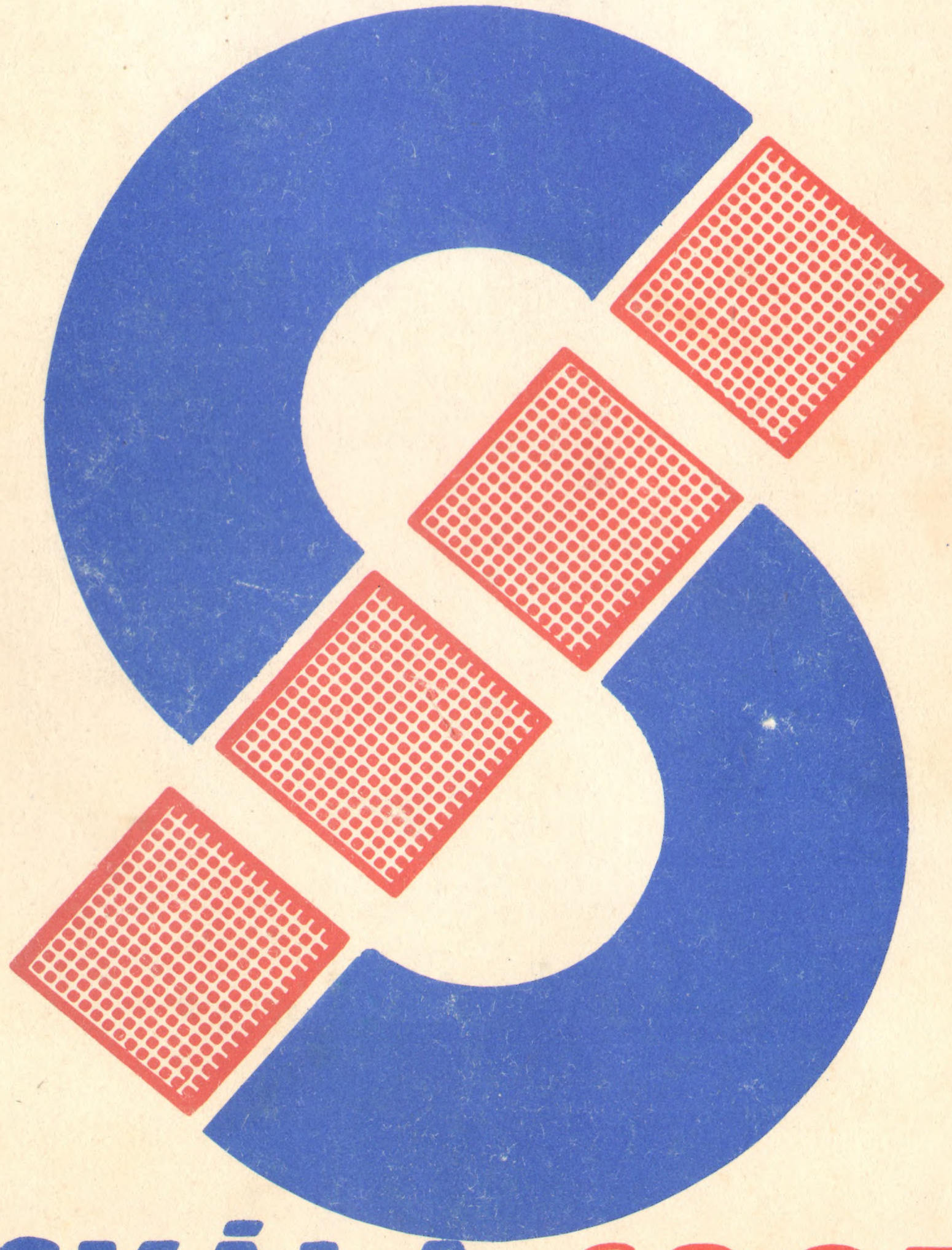
V. Fontosabb memóriahelyek 163

VI. Hangmagasság beállitása 167

VII. CP/M operációs rendszer használata 169

VIII. BASIC kulcsszavak, parancsok és röviditéseik . 171

commodore 64



SKÁLA·COOP